การนำเสนอแบบโต้ตอบสำหรับอาคาร 3 มิติ

INTERACTIVE PRESENTATION FOR 3D BUILDING

ชัชวาลย์ จักรแก้ว

ณัฐเศรษฐ์ ไตรทิพย์เจริญชัย

อภิญญา เลิศอารมย์รัตน์

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2553

ปริญญานิพนธ์ปีการศึกษา 2553

สาขาวิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง การนำเสนอแบบโต้ตอบสำหรับอาคาร 3 มิติ

INTERACTE PRESENTATION FOR 3D BUILDING

ผู้จัดทำ

1. นายชัชวาลย์ จักรแก้ว รหัสนักศึกษา 50010337
2. นายณัฐเศรษฐ์ ไตรทิพย์เจริญชัย รหัสนักศึกษา 50010519
3. นางสาวอภิญญา เลิศอารมย์รัตน์ รหัสนักศึกษา 50011857

_____ อาจารย์ที่ปรึกษา

(อาจารย์ชมพูนุช จินจาคาม)

_____ อาจารย์ที่ปรึกษา

(ดร. ปกรณ์ วัฒนจตุรพร)

การนำเสนอแบบโต้ตอบสำหรับอาคาร 3 มิติ

นายชัชวาลย์ จักรแก้ว 50010337
นายณัฐเศรษฐ์ ไตรทิพย์เจริญชัย 50010519
นางสาวอภิญญา เลิศอารมย์รัตน์ 50011857
อาจารย์ชมพูนุช จินจาคาม อาจารย์ที่ปรึกษา
ดร. ปกรณ์ วัฒนจตุรพร อาจารย์ที่ปรึกษาร่วม
ปีการศึกษา 2553

บทคัดย่อ

โครงงานนี้จัดทำขึ้นเพื่อศึกษา ออกแบบ และเขียนแอพพลิเคชั่นเกี่ยวกับอาคาร และสิ่งก่อสร้าง โดยเน้นใช้เทคโนโลยีการนำเสนอแบบโต้ตอบ(Interactive) และสามมิติโดยรูปแบบการพัฒนาแบบโปรแกรมแฟลช(Flash) ซึ่งสามารถนำมา ใช้งานได้กับทุกระบบปฏิบัติการ เพื่อใช้สำหรับการนำเสนอ เพื่อจุดประสงค์ต่างๆ ที่มีรูปแบบที่สวยงาม ใช้งานง่าย นำไปประยุกต์ใช้กับงานนำเสนอ(presentation) รวมถึงธุรกิจ อสังหา ริมทรัพย์ ซึ่งเป็นธุรกิจที่มีความ สำคัญอย่างมากในปัจจุบัน โดยตัวโปรแกรมจะเน้นในเรื่องของการ นำเสนอที่ใช้งานง่าย บุคคลทั่วไปก็ สามารถเข้ามาใช้งานได้เลยโดยไม่ต้องศึกษา ออกแบบด้วย ความสวยงาม เน้นแสดงออกถึงจุดยืนของแบนด์ (Brand) และ ความเป็นเอกลักษณ์ของแบนด์ รวมถึงเพื่อ ดึงดูดบุคคลให้เกิดความสนใจในตัวงานนำเสนอของอาคารอีกด้วย

ภายในโปรแกรมประกอบด้วยคำอธิบายโครงการเบื้องต้นพร้อมกับการใส่ภาพ และเสียงเพื่อสร้างความรู้สึกที่น่าสนใจ ในการนำเสนอที่เนื้อหาไม่มาก เน้นการใช้ภาพ และการจัดวางศิลป์เพื่อสื่อความหมายและรูปแบบทั้งหมดมากกว่าการ ใช้ตัวหนังสือ มีฟังก์ชั่นสำหรับการแสดงผลอาคารในโครงการทั้งหมดทั้งภายนอก ภายใน และบริเวณรอบๆตัวอาคาร สามารถทำการตอบโต้กับผู้ใช้ได้ทันทีเมื่อผู้ใช้สนใจในตึกนั้นๆ ก็สามารถเข้าไปดูได้ภายใน เป็นเชิงลึกขึ้น เพื่อดูแผนผังของชั้นและห้องรวมถึงผู้ใช้สามารถออกแบบภายในห้องนั้น สามารถใช้งานได้ในรูปแบบของภาพสามมิติ รวมถึงการปรับแต่ง เคลื่อนย้าย และ ออกแบบตำแหน่ง ของเฟอร์นิเจอร์ต่างๆได้ ส่งผลให้สามารถปรับแต่งรูป แบบที่พึ่งพอใจได้ตามความต้องการ ในห้องที่เลือกไว้ เพื่อความสะดวกต่อสายงานออกแบบภายใน(Interior) ในการออกแบบห้องต่อไป

Interactive Presentation For 3D Building

Ms. Chatchawan	Jakkaew	50010337
Mr. Nattasate	Trithipcharoenchai	50010519
Ms. Apinya	Lertaromrat	50011857
Mrs. Chompoonuch	Jinjakam	Advisor
Dr. Pakorn	Watanachaturaporn	Co-Advisor

Academic Year 2010

ABSTRACT

This project is intended for studying, designing and also creating the 3D appications of building and construction by using the interactive presentation and 3D. Using Flash platform which can work with every operation system(OS) for presentation purpose is to get the attractive the model, easy to use applied with the presentation including real estate which is play the important role on worldwide business. This program will focus on easy to presentation, everyone can use this program without learning it, and good designing focus on position of the brand and identity include attractive people to interest in the presentation of the building.

This program include the primary explanation of this project with how to transition the picture and the voice for created the taste to run the presentation by present not much content focus on using picture properly and the placement of art for meaning and a whole format more than using the alphabet. It's also has the function for display the building in whole project both exterior and interior design and also around the building which can response the user immediately and if the user interested in that building. The user can lock within a depth for lock the layout of the floor and also in the room. The user can interior decoration in 3D, adjust, move and design furniture and also have their own idea to design it to get their satisfaction in the room that has been chosen for convenience in interior design.

กิตติกรรมประกาศ

รายงานการนำเสนอแบบโต้ตอบสำหรับอาคาร 3 มิตินี้ คณะผู้จัดทำได้ศึกษา หาข้อมูล และเขียน แอพพลิเคชั่นเกี่ยวกับการนำเสนออาคาร 3 มิติให้มีการโต้ตอบ และตอบสนองกับผู้ใช้ ให้ผู้ใช้สามารถใช้งานโปรแกรมได้อย่างสะดวก มีการใช้งานที่เข้าใจง่าย คณะผู้จัดทำได้รับความแนะนำช่วยเหลือดูแลตลอดการศึกษาเป็นอย่างดีจาก อ.ชมพูนุช จินจาคาม และ ดร.ปกรณ์ วัฒนจตุรพร อาจารย์ที่ปรึกษา รวมทั้งอาจารย์วัจนพงศ์ เกษมศิริ ที่ให้ความช่วยเหลือจัดหาแนะนำเอกสารที่เกี่ยวกับการทำรายงานแก่คณะผู้จัดทำ ทำให้ปริญญานิพนธ์นี้สำเร็จลงได้ด้วยดี จึงขอกราบขอบพระคุณมา ณ ที่นี้ด้วย

ชัชวาลย์ จักรแก้ว
ณัฐเศรษฐ์ ไตรทิพย์เจริญชัย
อภิญญา เลิศอารมย์รัตน์

สารบัญ

หน้า

บทคัดย่อภาษาไทย.....I

บทคัดย่อภาษาอังกฤษ.....II

กิตติกรรมประกาศ.....III

สารบัญ.....IV

สารบัญรูป.....VI

บทที่ 1 บทนำ.....1

1.1 ความเป็นมา.....1

1.2 ความมุ่งหมายและวัตถุประสงค์.....1

1.3 ประโยชน์ที่คาดว่าจะได้รับ.....1

1.4 ขอบเขตของโครงงาน.....1

1.5 ขั้นตอนการดำเนินงาน.....2

บทที่ 2 พื้นฐาน ActionScript2.....3

2.1 ประวัติ Action Script.....3

2.2 เริ่มต้นเขียน Action Script.....4

2.3 คอนเซปต์คลาสเบื้องต้นในการเขียน Action Script.....6

2.4 พื้นฐานและแนวคิดของภาพเคลื่อนไหวต่างๆ คลาส และโปรแกรมเชิงวัตถุ.....13

2.5 Target Path.....17

2.6 Method.....25

2.7 Properties.....30

2.8 Events.....33

บทที่ 3 โมเดล 3D และการนำมาใช้.....43

3.1 การสร้างโมเดล 3D.....43

สารบัญ (ต่อ)

หน้า

3.2 วิธีการนำเข้าโมเดล .3DS ด้วย preFab และ Away3D.....46
3.3 การนำข้อมูลจากภายนอกมาใช้อย่างมีประสิทธิภาพโดยคำนึงถึงขนาดและความเร็ว..51

บทที่ 4 พื้นฐาน Away3D.....52
4.1 สร้าง 3D โปรเจค.....53
4.2 มุมมอง ฉาก และกล้อง.....62
4.3 พื้นฐานของโมเดล และ Sprites.....104

บทที่ 5 แอพพลิเคชั่น.....123
5.1 แอพพลิเคชั่นการนำเสนอแบบอาคาร และห้อง.....123
5.2 แอพพลิเคชั่นของห้อง 3D.....132

บทที่ 6 สรุป.....137
6.1 สรุป.....137
6.2 ปัญหาและอุปสรรค.....137
6.3 แนวทางการแก้ไข.....137
6.4 แนวทางการพัฒนาต่อ.....138

บรรณานุกรม.....139

สารบัญรูป

รูป		หน้า
2.1	เปรียบเทียบการเขียนโค้ดบน instance และบนเฟรม	6
2.2	การเชื่อมโยงของคลาส	7
2.3	การสร้างวัตถุใหม่ โดยสืบทอดมาจากคลาส	8
2.4	การกำหนดชื่อให้กับ instance	8
2.5	การตั้งชื่อให้ symbol	9
2.6	หน้าต่างของ Built-in classes	12
2.7	แผนผังลำดับเป็นต้นไม้	21
2.8	อธิบายการทำงานความสัมพันธ์ของคลาสและวัตถุต่างๆ	29
2.9	ตัวอย่าง method ของ MovieClip	30
2.10	การกำหนด Events บนตัว Instance	34
3.1	ลักษณะของโปรแกรม 3ds Max	44
3.2	ส่วนประกอบของ Viewport	44
3.3	หน้าต่างของ Viewport Configuration	46
3.4	แถบเครื่องมือใน 3ds Max	46
3.5	หน้าต่างโปรแกรม preFab	47
3.6	อธิบายวิธีการ Import ไฟล์	47
3.7	โมเดลที่โหลดเข้ามาใน preFab	48
3.8	การโหลดพื้นผิวเพื่อให้มาปรากฏบนวัตถุ	48
3.9	หน้าจอแสดง UV editer	49
3.10	หน้าต่างที่ใช้จัดการระหว่าง face กับ picture	49
3.11	ทำการ export ไฟล์ ไปเป็นนามสกุล .as	50
3.12	การ save ไฟล์ โดยระบุชื่อและ package ของคลาส	50
3.13	เมื่อทำการ export ตามขั้นตอนจะได้ไฟล์ .as และ โพล์เดอร์	50
4.1	อธิบายองค์ประกอบทั้งหมดที่ใช้งานในงานสามมิติ	52
4.2	ด้านหน้าของลูกบาศก์ ที่ได้จากการคอมไพล์	57
4.3	ภาพทรงกลมที่มีแสงและเงาปรากฏอยู่ด้านบน	60

สารบัญรูป (ต่อ)

รูป	หน้า
4.4 viewport ที่จะไม่เปลี่ยนแปลงตามขนาดของวัตถุ	63
4.5 Away3D ระบบพิกัดภาพซ้าย เมื่อเทียบกับระบบพิกัดพื้นฐาน Flash 10 ภาพขวา	65
4.6 จุด P ตำแหน่งที่ X = 100, Y = 100, Z = 50 ระบบพิกัดใน Away3D	66
4.7 สมการทฤษฎี Thales	67
4.8 สมการทฤษฎี Thales ตามสมการ 4.3	67
4.9 ความสัมพันธ์ของสมการ	68
4.10 ปัญหา z – sorting	69
4.11 ลำดับการทำงานของ painter's algorithm	70
4.12 สามเหลี่ยมที่มีระยะทางของภาพแตกต่างกัน	71
4.13 ภาพที่คาดหวังไว้และภาพที่ใช้ painter's algorithm	71
4.14 การแบ่งย่อยของภาพสามเหลี่ยม B	72
4.15 การแบ่งย่อยภาพสามเหลี่ยมและจัดอันดับใหม่	72
4.16 เปรียบเทียบการตั้งค่าจุดศูนย์กลาง	77
4.17 เปรียบเทียบการแสดงผลเมื่อมีการกำหนด viewport	78
4.18 เปรียบเทียบลำดับคลาสระหว่าง Away3D และ flash	79
4.19 การเปลี่ยนจุดในระบบพิกัดจะส่งผลต่อทุกแกน	84
4.20 การหมุนของแกนที่ส่งผลกระทบต่อกัน	85
4.21 สิ่งที่กล้องในสามมิติเห็น	89
4.22 การฉายจุดลงใน viewport และ การ render ภาพเพื่อให้ผู้ใช้เห็นบนหน้าจอ	89
4.23 ความสัมพันธ์ระหว่าง focus field of view และความสูงของ viewport	90
4.24 ความสัมพันธ์ระหว่าง focus field of view และความสูงของ viewport เมื่อเพิ่ม focus	91
4.25 การซูมเพื่อขยายหรือลดขนาดวัตถุ	91
4.26 ความแตกต่างระหว่างกล้องแบบ Target camera และ Free camera	92
4.27 เปรียบเทียบกล้องในฉากเดียวกันจากสี่ตำแหน่งกล้องที่แตกต่างกัน	94
4.28 ตัวอย่างหลังจากได้ทำการหมุนกล้องไป 10 องศา	96
4.29 เปรียบเทียบฉากเดียวกันกับการตั้งค่าการซูมและโฟกัสที่แตกต่างกันในกล้อง	98

สารบัญรูป (ต่อ)

รูป	หน้า
4.30 หลังจากหมุนกล้องไปด้านบน และด้านขวา โดยใช้ lookAt()	100
4.31 คุณสมบัติ panAngle และ tiltAngle	104
4.32 อธิบายการเกิด faces สามเหลี่ยมและตาข่ายสามเหลี่ยมจากจุด	106
4.33 สี่เหลี่ยมที่ถูกวาดสองมิติ แต่แสดงผลลัพธ์เป็นสามมิติ	107
4.34 ระนาบซึ่งตั้งค่า segmentsW และ segmentsH ไว้ที่ 10	111
4.35 เปรียบเทียบการตั้งค่า segmentsW segmentsH ของทรงกลม	114
4.36 ลูกบาศก์ที่สร้างโดยการใช้งานของ wire	118
4.37 การสร้างเส้นแบบสุ่มจากตัวอย่าง LinesInSpaceWithLineSegment	119
4.38 การสร้างรูปทรงต่างๆ ที่ใช้คลาส RegularPolygon class	121
5.1 หน้า Screen Server	123
5.2 Screensaver เมื่อมีภาพประกอบ	125
5.3 หน้า Home	125
5.4 หน้า Gallery	126
5.5 หน้าของแอพพลิเคชั่นที่แสดงบรรยากาศโดยรวม	127
5.6 หน้าของแอพพลิเคชั่นที่แสดงบรรยากาศภายในห้อง	127
5.7 แอพพลิเคชั่นในหน้าของการนำเสนออาคาร	128
5.8 แอพพลิเคชั่นจะแสดงภาพตึกที่ผู้ใช้สนใจ	128
5.9 ชั้นที่ผู้ใช้สนใจ	129
5.10 Floor Plan ในชั้นที่ 3 ตึก C	129
5.11 Floor Plan ในชั้นที่ 7 ตึก D	130
5.12 Room Plan ของห้องชนิด SD ขนาด 92 ตารางเมตร ซึ่งเป็นห้องเชื่อมสองชั้น	130
5.13 Fact Sheet	131
5.14 Project Info	131
5.15 หน้าหลักของการตกแต่งห้อง 3D	132
5.16 โมเดลที่มีการปรับเปลี่ยนมุมมอง โดยการหมุน และ zoom เข้า	133
5.17 มุมมองด้านบนที่ตั้งไว้เป็นมุมมองพื้นฐาน	133

สารบัญรูป (ต่อ)

รูป	หน้า
5.18 การหมุนเฟอร์นิเจอร์ และลบเฟอร์นิเจอร์ โดยเลือกที่ทีวี	134
5.19 การจัดการเฟอร์นิเจอร์ โดยเครื่องมือที่ชื่อ manage	134
5.20 หน้าต่างในการเลือก Material	135
5.21 หน้าต่างในการเลือกสี	135
5.22 หน้าต่างที่ใช้ในการเพิ่มเฟอร์นิเจอร์	136
5.23 หน้าต่างในการตั้งค่าขนาดของห้อง	136

บทที่ 1
บทนำ

1.1 ความเป็นมาของปัญหา

เนื่องจากในปัจจุบันเทคโนโลยี 3 มิติ เป็นที่น่าจับตามอง และมีความน่าสนใจเป็นอย่างมาก มีการนำเทคโนโลยีต่างๆ เข้ามามีบทบาทในชีวิตประจำวันเป็นอย่างมาก และเรื่องของภาพ 3 มิติก็มีความสวยงาม และความน่าดึงดูดในตัวเองเป็นทุนเดิมอยู่แล้ว หากนำมาประยุกต์ใช้กับแอพพลิเคชั่นที่ไว้ใช้สำหรับนำเสนองานขายได้ ก็สามารถทำให้เนื้องานมีความดึงดูด มีความเหมือนจริง และน่าสนใจได้มากขึ้น

การนำเอาเรื่องของภาพสามมิติมาทำแอพพลิเคชั่นสำหรับอาคารในปัจจุบันก็มีความน่าสนใจเป็นอย่างมาก เพราะธุรกิจอสังหาริมทรัพย์นั้น จำเป็นต้องใช้งานสามมิติ เพื่อสร้างอาคารในจินตนาการที่สมจริงขึ้นมาก่อนที่อาคารจริงจะสร้างขึ้นมาเป็นตัวดึกอย่างสมบูรณ์ เพื่อให้ผู้บริโภคสามารถตัดสินใจได้ง่ายขึ้น

1.2 วัตถุประสงค์ของโครงงาน

1) เพื่อสร้างแอพพลิเคชั่นที่สามารถนำเสนออาคารที่น่าสนใจใช้งานง่าย และสะดวกต่อการใช้งาน
2) เพื่อสร้างแอพพลิเคชั่นที่สามารถออกแบบภายในห้องเป็น 3 มิติ ตามที่ผู้ใช้ต้องการได้

1.3 ประโยชน์ที่คาดว่าจะได้รับ

1) ได้รับความรู้ และสามารถสร้างภาพพื้นฐาน 3 มิติได้
2) สามารถสร้างแอพพลิเคชั่นที่ใช้ออกแบบห้อง 3 มิติได้
3) สามารถนำความรู้ที่ได้รับ มาประยุกต์ใช้กับงาน 3 มิติประเภทอื่นๆได้

1.4 ขอบเขตของโครงงาน

สามารถสร้างแอพพลิเคชั่นนำเสนออาคารได้อย่างสวยงาม ใช้งานง่าย โดยเขียนจากโปรแกรมแฟลช และ เฟล็กส์บิวเดอร์(flex builder) การจัดวางองค์ประกอบทั้งหมด รวมถึงการออกแบบ ให้มีความน่าดึงดูด โดยเน้นรูป แบบที่เรียบง่าย สวยงาม และออกแบบเพื่อให้ใช้งานง่าย สำหรับบุคคลทุกประเภท โดยไม่ต้องศึกษาก่อนใช้งาน และออกแบบมาให้ใช้สำหรับจอสัมผัส(touch screen)ด้วย

ตัวแอพพลิเคชั่นสามารถใช้ออกแบบภายในห้องได้ตามความต้องการของผู้ใช้ในรูปแบบของภาพ 3 มิติรวมทั้งการขยับเคลื่อนย้ายเฟอร์นิเจอร์ การหมุนเฟอร์นิเจอร์ได้ และแอพพลิเคชั่นสามารถติดตั้งได้โดยอัตโนมัติเหมือนโปรแกรมโดยทั่วไป และสามารถลงแอพพลิเคชั่นนี้ได้ในมือถือบางประเภทได้

1.5 ขั้นตอนการดำเนินงาน

1) ศึกษาการเขียนโปรแกรมแฟลชการใช้คำสั่งแฟลช
2) หาข้อมูล และรูปภาพต่างๆ ที่จะนำมาใช้ในแอพพลิเคชั่น
3) เขียนโปรแกรม เพื่อสร้างแอพพลิเคชั่นสำหรับนำเสนออาคาร
4) ศึกษาการนำภาพ 3 มิติ มาใช้ในแอพพลิเคชั่น ศึกษาภาษาที่ใช้ในการสร้างแอพพลิเคชั่น
5) เขียนโปรแกรมเพื่อสร้างแอพพลิเคชั่นในการตกแต่งห้อง

บทที่ 2
พื้นฐาน ActionScript 2

แฟลช และแอกชันสคิปท์(ActionScript) เข้ามามีบทบาททั้งในการทำเว็บไซต์ โปรแกรมการนำเสนอ โปรแกรมการใช้งานภายในองค์กร สื่อการเคลื่อนไหว (Motion Graphic) การ์ตูน และเกม ถึงวันนี้ภาษา และ แอกชันสคิปท์(ActionScript) ยังคงให้การพัฒนาและให้ความสนใจจากผู้พัฒนาในหลายๆ สาขาอาชีพทั่วโลก ผู้ที่สนใจจะเข้ามาสู้สาขาวิชาชีพที่กล่าวมานั้นแทบจะปฏิเสธไม่ได้ที่จะใช้แฟลชเป็นเครื่องมือส่วนหนึ่งในการสร้างสรรค์ผลงานออกมา

แฟลชมีความสามารถทางด้านกราฟฟิกผสมผสานในเชิงเวคเตอร์ และแบบบิทแมพได้อย่างลงตัว เพื่อให้ผู้ใช้สามารถสร้างผลลัพธ์เชิงภาพได้อย่างยอดเยี่ยม และ แอกชันสคิปท์ ก็เป็นภาษาคอมพิวเตอร์ที่เข้าใจง่าย และมีความสามารถอย่างมากมาย ประกอบกับเป็นภาษาที่อยู่ในโปรแกรมแฟลชเอง ทำให้การทำงานระหว่างความสวยงาม และลูกเล่นโปรแกรมผสมรวมกันอย่างลงตัวในโปรแกรมตัวนี้

2.1 ประวัติภาษาแอกชันสคิปท์

ภาษาแอกชันสคิปท์ (เรียกย่อๆว่า เอเอส(AS)) นั้นเป็นภาษาที่ใช้งานกับไฟล์แฟลชเป็นหลัก แต่สามารถเขียนพัฒนาได้จากหลายโปรแกรมอื่นๆในปัจจุบัน เช่น โปรมแกรมแอร์(AIR) (เน้นพัฒนาแอปปริเคชันบนเดสทอป) โปรแกรมเฟล็กส์บิวเดอร์(แฟลชบิวเดอร์เวอร์ชัน 4 ในปัจจุบัน) หรือโปรแกรมโอเพ่นซอรส์ (Open Source หมายถึงโปรแกรมที่ไม่มีการปิดบังโค้ด(code) เพื่อให้คนอื่นสามารถร่วมพัฒนาต่อยอดได้ และที่สำคัญคือเป็นฟรีโปรแกรม) อย่าง แฟลชดิวีลอปเปอร์ (flashdeveloper) หรือ อื่นๆได้เช่นกัน โดยประโยชน์ของภาษาแอกชันสคิปท์นั้น คือการช่วยให้ผู้ชมแฟลชสามารถทำการอินเตอร์แอคทีพหรือตอบโต้กับวัตถุต่างๆได้ นอกจากนี้ เมื่อภาษาแอกชันสคิปท์ก้าวกระโดดพัฒนาไปมากขึ้น การเชื่อมต่อกับแหล่งข้อมูลอื่นๆ ก็ทำได้มากขึ้น รวมถึงบางสิ่งที่แฟลชต้องอาศัยภาษาอื่นช่วยในการทำ ก็สามารถทำได้ด้วยตัวมันเองในภายหลัง

สำหรับแฟลชในเวอร์ชันยุคแรกเริ่มนั้น เวอร์ชันที่ใช้คือ 1.0 ซึ่งการทำงานนั้นก็ค่อนข้างจะครบถ้วนสมบูรณ์ระดับหนึ่ง ในสมัย flash 5 และทำงานได้มากขึ้นใน เวอร์ชัน 6 หรือ MX

แต่ทั้งนี้กระแสการเขียนโปรแกรมเชิงวัตถุ (Object Oriented Programming) เป็นการเขียนโปรแกรมโดยมองสิ่งต่างๆเป็นวัตถุที่มี คุณสมบัติ การดำเนินการต่างๆ ฯลฯ) นั้นเป็นสิ่งที่ขาดเสียไม่ได้เลยที่จะมองข้าม ถึงแม้ว่าภาษาแอกชันสคิปท์ 1.0 นั้นจะเขียนโปรแกรมเชิงวัตถุได้ก็ตาม แต่ไม่ใช่รูปแบบการเขียนโปรแกรมเชิงวัตถุที่เป็นมาตรฐานทั่วไปสักเท่าไรนัก (AS 1.0 นั้นใช้คำสั่ง

โปรโตไทป์(prototype) ที่แปลว่าต้นแบบ ในการสร้างคลาส (Class) ดังนั้น AS 2.0 จึงได้ถูกนำมาใช้ โดยสามารถเขียนภาษาแอกชันสคิปท์เป็นไฟล์สำหรับคลาสแยกออกมาต่างหาก โดยมีการจัดวางรูปแบบการเปิดและปิด การประกาศตัวแปรต่างๆ ให้ตรงตามลักษณะโปรแกรมเชิงวัตถุมากขึ้น

เมื่อ flash cs3 ภายใต้แบรนด์ของ Adobe ถูกปล่อยออกมา ความเป็นโปรแกรมเชิงวัตถุสมบูรณ์มากขึ้นของ AS ก็ถูกปล่อยออกมาในนาม AS 3.0 ที่ค่อนข้างจะเปลี่ยนรูปแบบการเขียนแบบเดิมๆ ของ AS ไปเลย และ การอ้างอิงวัตถุ ตัวแปร หรือฟังก์ชันต่างๆ ก็ค่อนข้างจะต้องรัดกุมมากขึ้น เพราะมีข้อบังคับ ของประเภทข้อมูลอย่างเช่นตัวแปรและอื่นๆ มากขึ้น นอกจากนี้การประกาศคำสั่งอีเวนต์ เพื่อตรวจสอบว่าวัตถุนั้นๆควรจะทำงานเมื่อใดก็เปลี่ยนไป โดยไม่สามารถใช้รูปแบบการเขียนแบบเก่ามาผสมได้อีก (จริงๆแล้ว รูปแบบนี้มีการใช้มาก่อนแล้วในเวอร์ชันก่อนหน้านี้ โดยเฉพาะ component ต่างๆ เพียงแต่มีการเปลี่ยนจากการระบุค่าสตริง (String) ไปเป็นพรอพเพอร์ตี้จากคลาสอีเวนต์แทน เพื่อให้โปรแกรมสามารถตรวจสอบได้ว่ามีการส่งค่าถูกต้องหรือไม่)

2.2 เริ่มต้นเขียนภาษาแอกชันสคิปท์

2.2.1 อัลกอริธึม (Algorithm)

อัลกอริธึมนั้นจะหมายถึงลำดับขั้นตอนการทำงาน โดยอัลกอริธึมนั้นเป็นเรื่องของประสบการณ์ของแต่ละคน เนื่องจากการเขียนโปรแกรมไม่มีหลักการตายตัว บางคนอาจเขียนให้โปรแกรมทำงานได้ตามที่ต้องการ แต่โค้ดยาว ลำดับขั้นตอนเยอะ ซับซ้อน แต่บางคนเขียนโค้ดได้ผลลัพธ์เดียวกัน แต่โค้ดสั้นกระชับ ลำดับขั้นตอนการทำงานน้อยมาก ซึ่งเรื่องเหล่านี้ ก็อยู่บนพื้นฐานความเข้าใจในการใช้งานคำสั่ง และพื้นฐานการจัดวางโครงสร้างของโปรแกรมเช่นเดียวกัน นอกจากนี้ ยังมีเรื่องอื่นๆให้ศึกษาอีก อาทิเช่น Frame Work ซึ่งเกี่ยวกับรูปแบบที่จัดเอาไว้สำหรับการเขียนโปรแกรมรวมไปถึงคำสั่งต่างๆ ที่จัดเตรียมมาให้ใช้งานได้สะดวกมากขึ้น และ Design Pattern ซึ่งเกี่ยวข้องกับการจัดวางโครงสร้างของโปรแกรมให้ได้มาตรฐาน

2.2.2 วิธีการเขียนภาษาแอกชันสคิปท์

การเขียนภาษาแอกชันสคิปท์นั้นจะแบ่งออกเป็น 3 รูปแบบคือ

1) การเขียนแบบ Script Assist ซึ่งจะเป็นการกรอกข้อมูลลงในช่องแบบฟอร์มที่กำหนดมาให้ (ซึ่งเคยถูกตัดทิ้งไปในเวอร์ชัน MX และนำกลับมาใหม่)
2) การเขียนภาษาแอกชันสคิปท์แบบ advance หรือแบบที่เขียนโค้ดเองทั้งหมด
3) การเขียนภาษาแอกชันสคิปท์โดยใช้ พาเนล Behavior ซึ่งจะเป็นการกำหนดโค้ดให้กับวัตถุโดยใช้วิธีการเลือกคำสั่งต่างๆ แบบรวดเร็ว และจะปรากฏคำสั่งใน Actions Panel

2.2.3 พื้นที่ในการเขียนภาษาแอกชันสคิปท์

สามารถเขียนภาษาแอกชันสคิปท์ได้ทั้งหมด 3 ที่หลักๆคือ

1) เขียนไว้บนเฟรม(Frame)โดยทำการคลิกที่เฟรมใดๆ แล้วก็เขียนโค้ดลงไป
2) เขียนไว้บนตัวอินสแทนซ์(Movie clip หรือ Button) โดยคลิกที่ตัวอินสแทนซ์ที่ต้องการแล้วใส่โค้ดลงไป โดยมีเงื่อนไขพิเศษคือ ต้องทำการกำหนดอีเวนต์(event) ก่อนเสมอมิฉะนั้นโปรแกรมจะแจ้งเตือน error
3) เขียนโค้ดไว้ในไฟล์นามสกุล as ซึ่งวิธีการนำมาใช้นั้นมีหลายรูปแบบ ทั้งการใช้คำสั่ง include เพื่อดึงเข้ามา หรือถ้าเขียนไว้ในรูปแบบของคลาส จะใช้คำสั่งอิมพอร์ต

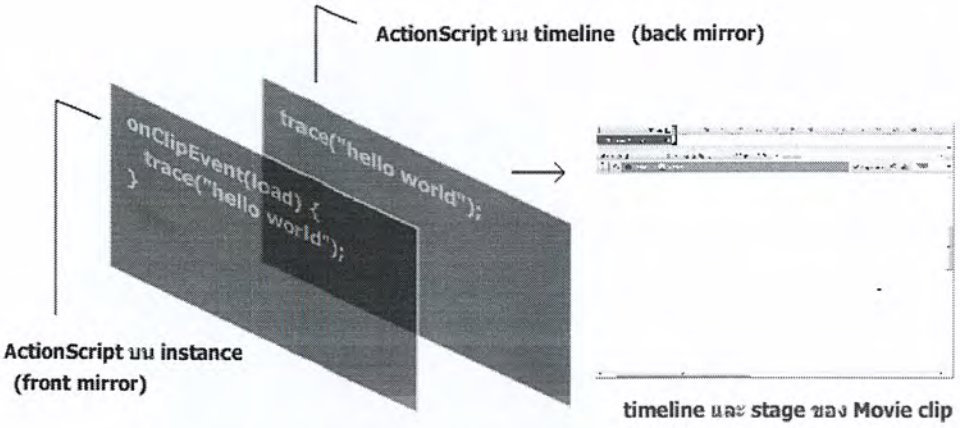
นอกจากนี้ยังสามารถนำไปกำหนดให้กับ document (AS3) หรือ มูฟวี่คลิปในไลบรารีได้ด้วย โดยจะมีช่องชื่อว่าคลาสเพื่อให้กำหนด as ไฟล์ลงไป

2.2.4 ข้อแตกต่างระหว่างการเขียนโค้ดบนอินสแทนซ์และบนเฟรม

เพื่อให้เห็นภาพมากขึ้นในการเขียนโค้ดบนเฟรม กับการเขียนโค้ดบนตัวอินสแทนซ์จะยกตัวอย่างดังนี้

ในที่นี้จะเปรียบเทียบว่ามูฟวี่คลิป ให้เป็นเสมือนกระจกใสแผ่นนึง เมื่อคลิกเข้าไปในมูฟวี่คลิปตัวนั้น ถึงจะเจอไทม์ไลน์และ stage ของมูฟวี่คลิป ฉะนั้นการเขียน โค้ดบนตัวมูฟวี่คลิปก็คือการเขียนโค้ดลงบนด้านหน้ากระจก ส่วนการเขียนโค้ดลงบนเฟรมของไทม์ไลน์ของมูฟวี่คลิปก็คือการเขียนโค้ดลงบนหลังกระจก ฉะนั้นแม้ว่าคุณจะเขียนโค้ดลงบนตัวอินสแทนซ์หรือ บน frame ของมูฟวี่คลิปก็ตาม การเขียนโค้ดทั้ง 2 ตำแหน่งนี้ถือว่าอยู่บนกระจกแผ่นเดียวกัน ซึ่งคุณสามารถอ้างอิงถึงอินสแทนซ์บน stage ของอินสแทนซ์ตัวที่ถูกเขียนโค้ดได้ (ถือว่าเป็นส่วนที่อยู่ติดกัน)

แต่สำหรับกรณีของการอิงคำสั่งจะมีข้อควรรู้อย่างนึงก็คือ การทำงานด้านหน้ากระจกจะทำก่อน ด้านหลังกระจก ฉะนั้นถ้าคุณมีการประกาศคำสั่งลงบนด้านหลังกระจก คำสั่งบนด้านหน้ากระจกจะยังไม่สามารถอ้างอิงคำสั่งบนด้านหลังกระจกได้ใน ทันที แต่จะสามารถทำงานได้ในครั้งถัดไปที่คำสั่งถูกทำงานไปแล้วหนึ่งครั้ง (แต่กรณีการอ้างอิงถึงอินสแทนซ์นั้นไม่เกี่ยวเนื่องจากว่าลำดับการทำงานของโปรแกรม จะประมวลผลในส่วนของกราฟฟิคต่างๆก่อน จึงสามารถอ้างอิงได้ในทันที) จากรูปจะแยกเป็นกระจกแผ่นหน้าและแผ่นหลัง เพื่อให้เห็นความแตกต่างระหว่างการเขียนโค้ดทั้งสองที่ แต่ที่จริงแล้วก็เป็นกระจกแผ่นเดียวกัน



รูป 2.1 เปรียบเทียบการเขียนโค้ดบนอินสแทนซ์และบนเฟรม

สำหรับแฟลชนั้นการเขียนคำสั่งสามารถทดสอบผลลัพธ์และแสดงผลด้วยคำสั่ง trace ได้ทางหน้าต่าง output ภายในโปรแกรมแฟลชได้เลย และกรณีที่มีคำสั่งส่วนใดเกิดข้อผิดพลาด ก็จะแสดงผ่านทางหน้าต่าง Compiler Errors ซึ่งกรณีที่เป็นภาษาทาง server side script อย่างเช่น php จะใช้คำสั่งว่า echo ซึ่ง echo ในที่นี้ใช้ทั้งตรวจสอบค่า และแสดงผลลัพธ์ออกทางบราวเซอร์ได้เช่นเดียวกัน ข้อแตกต่างก็คือ คำสั่ง trace นี้จะไม่ถูกแสดงใน swf หรือเมื่อทดสอบผ่านทางบราวเซอร์นั่นเอง (แต่ก็มี plugin สำหรับ firefox เพื่อตรวจสอบค่าที่ trace ออกมาได้)

การเขียนโปรแกรมที่ดี ควรตรวจสอบได้ว่า คำสั่งตรงไหนที่น่าจะผิดพลาด ตรงไหนทำ ตรงไหนไม่ทำ ตรงไหนมีค่า ตรงไหนไม่มีค่า ฉะนั้นควรใช้ trace เพื่อตรวจสอบให้เป็นนิสัย ซึ่งจะช่วยให้พบปัญหาได้ง่ายขึ้น

2.3 คอนเซปต์คลาสเบื้องต้นในการเขียนภาษาแอกชันสคิปท์

การเรียนรู้และทำความเข้าใจคอนเซปต์และคำนิยาม โดยจะอธิบายความเข้าใจในเรื่องคลาสเบื้องต้น (ซึ่งคอนเซปต์ในเชิงมาตรฐานอาจจะต่างไปจากนี้ในบางจุด เนื่องจาก AS2 ยังไม่ถือว่าเป็นการเขียนโปรแกรมเชิงวัตถุแบบสมบูรณ์สักเท่าไรนัก)

ในการเขียนโปรแกรมหรือสร้างงานแฟลชนั้นก็ไม่ต่างกัน คุณต้องสร้างองค์ประกอบต่างๆ ขึ้นมาเองทั้งหมด ตามที่ต้องการ

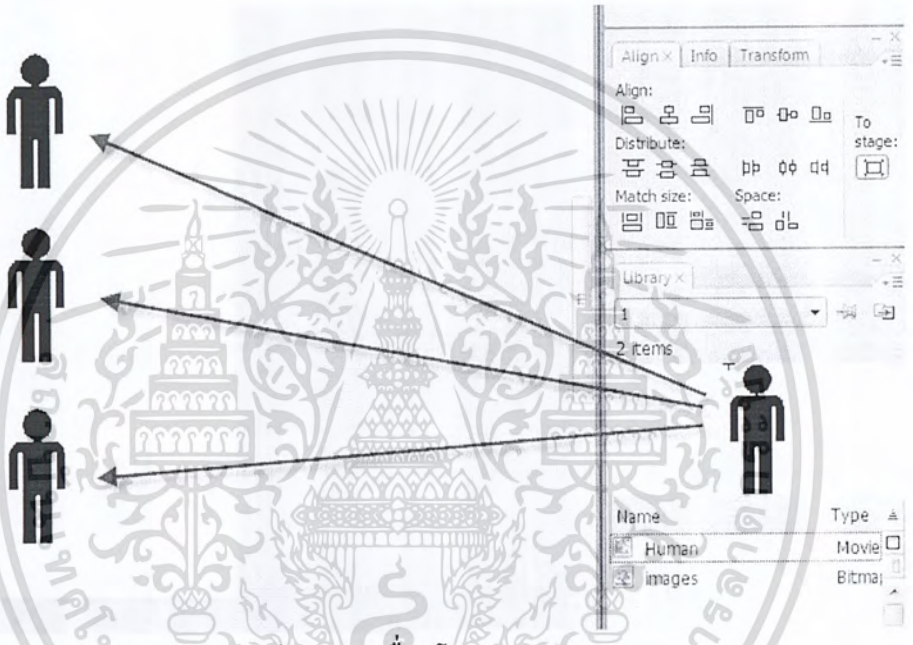
ทีนี้จะแบ่งวัตถุ สิ่งของ หรืออะไรก็ตาม จะแบ่งมันออกเป็น "คลาส (Class)" ซึ่งคลาสนั้นเปรียบเสมือนต้นแบบ ของวัตถุ ก็ว่าได้

2.3.1 คลาสที่เป็นรูปธรรม และนามธรรม

สำหรับคลาสนั้น เมื่อถูกสร้างออกไปใหม่ จะเรียกว่า วัตถุ (Object) หรือ ตัวแทน (Instance) และจะสามารถทำงานได้เหมือนกับคลาสทุกอย่าง

กรณีในแฟลชนั้น เมื่อสร้างซิมโบล์ขึ้นมา มันจะถูกเก็บอยู่ในไลบราลี และเมื่อทำการลากจากไลบราลีมาวางบน stage ก็ถือว่ามันได้ถูกสร้างขึ้นมาใหม่โดยลอกเลียนแบบทุกอย่างจากต้นแบบออกมา เรียบร้อยแล้ว และวัตถุที่สร้างใหม่นี้ เรียกว่าอินสแทนซ์แต่จะมองว่ามันคืออ็อบเจ็กก็ได้เช่นเดียวกันเพราะอันที่จริงมันคือสิ่งเดียวกัน ซึ่งอาจจะมองว่าซิมโบล์ ก็เปรียบเสมือน คลาส ที่เป็นรูปธรรม สามารถเห็นมันได้ เพราะมันมีรูปร่าง (shape)

เพื่ออธิบายการทำงานของคลาส และวัตถุที่สืบทอดมาจากคลาสนั้นๆ จะแสดงตัวอย่างดังภาพ คือคลาสแม่นั้น จะมีวัตถุที่สืบทอดมาก็คือคนที่อยู่ทางด้านซ้ายนั่นเอง



รูป 2.2 การเชื่อมโยงของคลาส

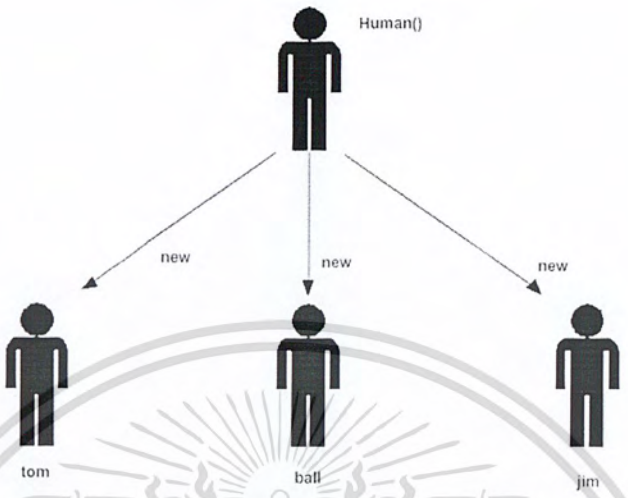
แต่ในการเขียนภาษาแอกชันสคิปท์หรือภาษาอื่นๆนั้น จะสร้างคลาสออกจะเป็นนามธรรม เพราะมันไม่มีรูปร่างให้เห็น แต่มันก็ยังสามารถทำงานได้

อย่างเช่นต้องการสร้างมนุษย์ (Human) ขึ้นมา ก็จะได้ คลาส Human ซึ่งเป็นต้นแบบหลักๆของมนุษย์ทั่วโลก และสามารถสร้างมนุษย์ออกมาหลายๆคน ด้วยกระบวนการที่ชื่อว่าคอนสตรัคท์ โดยรูปแบบการเขียนภาษาแอกชันสคิปท์จะเป็นดังตัวอย่าง 2.1

ตัวอย่าง 2.1

```
tom  = new Human();
ball = new Human();
jim  = new Human();
```

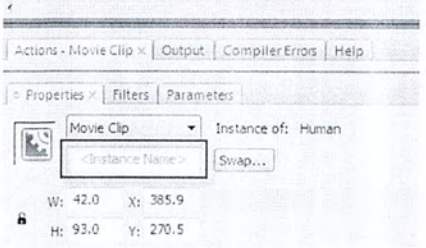
สังเกตุว่าชื่อ คลาสจะขึ้นต้นด้วยตัวอักษรใหญ่ และมี () ต่อท้าย (เพราะจริงๆแล้วมันคือรูปแบบของฟังก์ชัน(function) และจะเรียก Human นี้ว่าเป็นตัวคอนสตรัคเตอร์)



รูป 2.3 การสร้างวัตถุใหม่ โดยสืบทอดมาจากคลาส

2.3.2 การกำหนดชื่อให้กับอินสแทนซ์หรืออ็อบเจ็ก

สำหรับอินสแทนซ์บน stage อย่างมูฟวี่คลิป(Movie clip)และ Button สามารถกำหนดชื่อให้กับมัน เพราะถ้าไม่มีชื่อ ก็ไม่สามารถอ้างอิงมันได้ โดยการตั้งชื่อสำหรับมูฟวี่คลิปและ Button สามารถตั้งชื่อได้การคลิกเลือกอินสแทนซ์ตัวนั้นๆ แล้วกำหนดชื่อลงไปในช่อง Instance Name ที่พาเนลพรอพเพอร์ตี้



รูป 2.4 การกำหนดชื่อให้กับอินสแทนซ์

สำหรับกรณีการสร้างอ็อบเจ็ก(object) ด้วยการคอนสตรัคท์นั้น ชื่ออ็อบเจ็กดังกล่าว สามารถใช้อ้างอิง ได้เลยทันที จากตัวอย่างก่อนหน้านี้ก็คือชื่อ jim, ball, tom นั่นเอง

นอกจากนี้ Human ถือว่าเป็นชื่อคลาสที่สามารถอ้างอิงได้ แต่สำหรับกรณีซิมโบล์นั้น ชื่อซิมโบล์ถือว่าเป็นชื่อเพื่อให้ทราบว่าอินสแทนซ์ตัวนั้นถูกสร้างจากซิมโบล์ตัวใด แต่ยังไม่ใช่ชื่อที่ใช้ในการอ้างอิงสำหรับภาษาแอกชันสคิปท์ได้จริงๆ ซึ่งวิธีการตั้งชื่อให้สำหรับนั้น จะกำหนดผ่านซิมโบล์ทางไลบรารีโดยตั้งค่าผ่านช่อง linkage โดยคลิกเมาส์ขวาที่ซิมโบล์เลือกพรอพเพอร์ตี้แล้วกดปุ่ม Advance หรือเลือกเมนู linkage โดยตรง จะปรากฏส่วนกรอกชื่อ Identifier โดยเลือก Export for ActionScript ซึ่งโปรแกรมจะกรอกชื่อเดียวกับซิมโบล์มาให้อัตโนมัติในช่องง Identifier แต่สามารถแก้ไขได้

รูป 2.5 การตั้งชื่อให้ซิมโบล์

ไม่ว่าจะชื่ออินสแทนซ์หรือชื่อของ Identifier ไม่สามารถตั้งชื่อซ้ำกันได้ เพราะจะทำให้โปรแกรมไม่สามารถอ้างอิงได้ เนื่องจากไม่รู้จะอ้างอิงตัวไหน

กรณีที่เป็นภาษาแอกชันสคิปท์เวอร์ชัน 3.0 ไอเด็นติฟายเออร์(ActionScript 3.0 Indentifier นั้นจะถูกตัดออกไป แต่จะให้ทำการตั้งชื่อที่ช่องคลาสเพื่อให้ซิมโบล์นั้นสามารถทำการคอนสตรัคท์(construct) จากภาษาแอกชันสคิปท์ได้เลย ซึ่งอันที่จริงแล้วช่องคลาสนี้จะใช้เป็นชื่อเพื่ออ้างอิง หรือว่านำไฟล์คลาสที่ถูกเขียนในรูปแบบไฟล์ as มาผูกเอาไว้ก็ได้ เพื่อให้ซิมโบล์นั้น

ได้รับคุณสมบัติจากคลาสที่ระบุลงไป (as file) ซึ่งวิธีการระบุคลาสไฟล์นี้สามารถทำได้ในภาษาแอกชันสคิปท์เวอร์ชัน 2.0 เช่นเดียวกัน แต่ไม่สามารถทำได้ถึงการคอนสตรัคท์ด้วยภาษาแอกชันสคิปท์ยังคงต้องใช้ Identifier เป็นชื่ออ้างอิง

2.3.3 คุณสมบัติ (Properties)

เรื่องของคุณสมบัติ เป็นอะไรได้หลากหลายรูปแบบ ไม่ว่าจะเป็นคุณสมบัติภายนอก รวมถึงคุณสมบัติภายใน และอื่นๆ ทั้งหลาย ถ้าเอายกตัวอย่างของมนุษย์เลยก็แบ่งจำแนก กันเป็น สีผิว สูง ต่ำ ดำ ขาว สิ่งพวกนี้ถือว่าเป็นคุณสมบัติภายนอกที่เห็นได้ง่ายๆ แต่ก็ยังมีสิ่งอื่นที่เป็นคุณสมบัติได้อีก เช่น สัญชาติ เชื้อชาติ ศาสนา ต่างๆเป็นต้น

สำหรับกรณีมูฟวี่คลิปสามารถปรับค่า ความสูง(height) ความกว้าง(width) ความโปร่งใส่ (alpha) ได้ ซึ่งพวกนั้นก็ถือว่าเป็นคุณสมบัติของมูฟวี่คลิปที่ถูกกำหนดเอาไว้แล้ว

ซึ่งคุณสมบัติ ต่างๆเหล่านั้น สามารถที่จะอ้างอิงเพื่อตรวจสอบได้ว่าคุณสมบัตินั้นมีค่าเท่าไร และก็สามารถกำหนดค่าใหม่ได้ ซึ่งจะอยู่ในรูปแบบของ ตัวแปร(variable) แทน ซึ่งในเชิงคลาสแล้ว คุณสมบัติ นั้นก็เปรียบเสมือน ตัวแปรค่านึงที่อยู่ภายในคลาสนั่นเอง ซึ่งจะได้เรียนรู้เกี่ยวกับ คุณสมบัติ กันต่อไป แต่ขอให้เข้าใจความหมายในเบื้องต้นเสียก่อน

ตัวอย่างเช่น ความสูงของมูฟวี่คลิปมี instance name ชื่อ tom ก็จะสามารถอ้างอิงได้

2.3.4 การกระทำ (method)

คลาสมีความสามารถที่จะกระทำได้ เรียกสิ่งนี้ว่าเมธอด (method) อาทิเช่น มนุษย์สามารถที่จะเดิน กิน เล่น เป็นต้น การสั่งให้คลาสนั้นทำงาน จะมีรูปแบบดังตัวอย่าง 2.2

ตัวอย่าง 2.2

```
tom.eat()
```

นอกจากนี้ สามารถเพิ่มรายละเอียดอย่างเช่นให้กินอะไร โดยกำหนดหรือส่งค่าให้กับเมธอดนั้นได้ด้วยรูปแบบดังตัวอย่าง 2.3

ตัวอย่าง 2.3

```
tom.eat("hamburger");
```

ซึ่งในการเขียนโปรแกรมเชิงวัตถุ จะมีหลักการว่า 1 เมธอด สามารถทำงานได้หลายแบบ

และสำหรับเมธอดนั้นในเชิงคลาส จะมองว่า คือ ฟังก์ชัน (function) หรือรูปแบบคำสั่ง ที่อยู่ภายใน คลาสนั้นๆ ก็ได้

เมธอด นั้นจะได้รับการสืบทอด (Inheritance) มาจากคลาสอีกทีหนึ่ง ฉะนั้นคลาสทำอะไรได้ อ็อบเจ็กนั้นก็ทำได้เช่นเดียวกัน แต่ไม่ได้จำกัดว่าจะต้องทำได้เหมือนกันเสมอไป เพราะสามารถเพิ่มเมธอดใหม่ๆเข้าไปให้อ็อบเจ็กได้ หรือว่าเมธอดชื่อเดียวกันก็อาจจะไม่ต้องทำงานเหมือนต้นแบบเสมอไป (overriding หรือการทับ)

2.3.5 เหตุการณ์ (event)

เหตุการณ์หรืออีเวนต์นั้นอาจจะไม่ได้อยู่ในนิยามของคลาสโดยตรง อันที่จริงมันก็คือเมธอดเพียงแต่เป็นเมธอดที่ทำเมื่อเกิดเหตุการณ์ใดๆ นั่นเอง

เรื่องของอีเวนต์นั้นโดยเบื้องต้นจะเกี่ยวกับการตอบสนองต่อปฏิกิริยาใดๆหรือการ interactive อย่างเช่น ถ้าคุณสร้างปุ่มขึ้นมา 1 ปุ่ม แต่คุณไม่กำหนดอีเวนต์อะไรให้กับปุ่มนั้นๆเลย มันก็จะไม่สามารถที่จะตอบสนองกับผู้ใช้ได้ เมื่อกดก็ไม่ทำอะไร เมื่อปล่อยเมาส์จากการกดปุ่มก็ไม่ทำอะไร แต่ถ้ากำหนดอีเวนต์ลงไป มันก็จะตอบสนองการกระทำตามอีเวนต์ที่คุณกำหนดได้

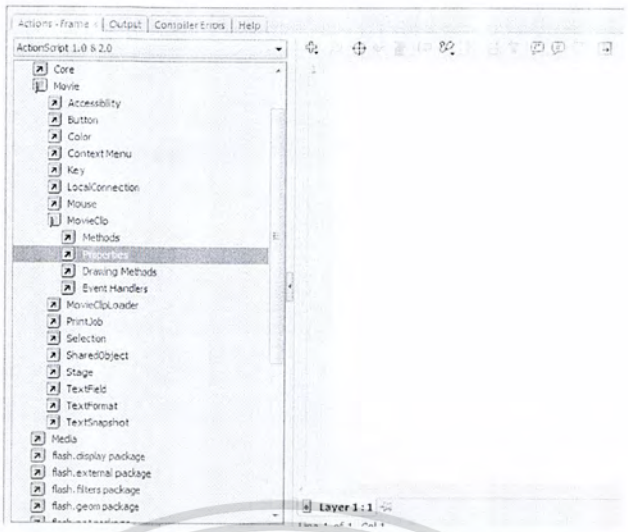
ตัวอย่างอื่นๆ เช่น การโหลดอะไรสักอย่าง เมื่อโหลดเสร็จ ก็ให้เกิดอีเวนต์บอกว่า โหลดเสร็จแล้วเป็นต้น แต่ในบางครั้ง คลาสบางอย่างอาจจะไม่มีอีเวนต์ที่ต้องการกำหนดมาให้ ก็ต้องทำการตรวจสอบจากสิ่งที่ต้องการทราบแล้วเรียกใช้เมธอดหรือ คำสั่งใดๆ เอาเอง อย่างเช่น การเล่นเพลง เหมือนเครื่องวิทยุ ในสมัยก่อน เมื่อเล่นจบเทป ก็จะเด้งปุ่มเล่น เพื่อแสดงให้รู้ว่า จบหน้าเทปแล้วนะ แต่ในโปรแกรมอาจจะไม่มีอีเวนต์นี้ ก็อาจจะต้องเขียนตรวจสอบเองว่า เล่นจบเพลงหรือยัง ถ้าจบแล้ว ก็ค่อยทำงานคำสั่งต่อไป

คลาสแต่ละอย่างนั้น มีอีเวนต์ที่แตกต่างกันออกไป และรูปแบบในการกำหนดอีเวนต์ก็มีหลายรูปแบบอีกด้วย

2.3.6 คลาสที่ถูกสร้างเอาไว้แล้ว (Built-in classes)

ปกติแฟลชจะมีคลาสที่ถูกเขียนมาให้แล้วบางส่วน ซึ่งผู้ใช้ไม่จำเป็นต้องเขียนเอง ก็สามารถนำมาใช้ได้เลย เรียกว่า built-in classes นอกจากนี้ ยังมีโปรแกรมเมอร์อีกมากมาย ที่เขียนคลาสมาให้ใช้อีกด้วย เพียงศึกษาคำสั่งต่างๆของคลาสนั้นๆว่าใช้ทำอะไรได้บ้าง

สำหรับคลาสที่ถูกสร้างมาให้แล้วสามารถที่จะเปิดดูได้ผ่านทาง help โดยการกด <F1> ได้ หรือจะลากครอบคำสั่งที่ต้องการแล้วกด <F1> เพื่อเปิดดูทันทีก็ได้ หรืออีกวิธีก็คือ สามารถดูได้ในส่วนของพาเนลทางด้านซ้ายของแอกชันพาเนล(Action Panels) ดังภาพ



รูป 2.6 หน้าต่างของ Built-in classes

จากภาพจะเป็นตัวอย่างของคลาสกลุ่มมูฟวี่คลิปซึ่งจะมีรายการ properties method event บอกเอาไว้ และถูกจัดแบ่งเป็นกลุ่มไว้อย่างละเอียด

ณ ตรงนี้ จะขอยกตัวอย่างคลาสคร่าวๆ เพื่อให้เห็นว่าแฟลชได้แบ่งคลาสสำหรับทำงานหน้าที่ส่วนใดไว้บ้าง ดังตัวอย่างคลาส 2.4

ตัวอย่างคลาส 2.4

Accessibility	เกี่ยวกับการเชื่อมต่อกับตัวอ่านหน้าจอ (screen readers) และอื่นๆ
Button	เกี่ยวกับปุ่ม
Color	เกี่ยวกับควบคุมสี
Content Menu	เกี่ยวกับเมนูเมื่อคลิกเมาส์ขวา
LocalConnection	เกี่ยวกับการเชื่อมต่อเพื่อทำงานร่วมกันระหว่าง swf ตัวอื่นๆ
Mouse	เกี่ยวกับการควบคุมเมาส์
MovieClip	เกี่ยวกับมูฟวี่คลิป
MovieClipLoader	เกี่ยวกับการโหลดไฟล์ต่างๆ
PrintJob	เกี่ยวกับการปริ้นต์
Selection	เกี่ยวกับการควบคุม text ที่ถูกแก้ไข
SharedObject	เกี่ยวกับบันทึกข้อมูลในส่วน local (เครื่องนั้นๆ)
Stage	เกี่ยวกับการควบคุมและดึงข้อมูลจาก Stage
TextFormat	เกี่ยวกับรูปแบบของข้อความใน Text Field
TextSnapshot	เกี่ยวกับการทำงานกับ static text ในมูฟวี่คลิป

สำหรับด้านบนเป็นแค่บางส่วนเท่านั้น ซึ่งสามารถดูคลาสทั้งหมดใน AS2.0 ได้โดยกด <F1> เพื่อเปิดการช่วยเหลือ แล้วไปยัง Book ชื่อ ActionScript 2.0 Languaged Reference และคลิกดูในส่วน ActionScript classes ซึ่งจะเป็นรายการ คลาสทั้งหมดที่สามารถใช้งานได้ รวมถึงคำอธิบายต่างๆ และตัวอย่าง

2.4 พื้นฐานและแนวคิดของภาพเคลื่อนไหวต่างๆคลาสและโปรแกรมเชิงวัตถุ

flash engine สร้างขึ้นจากระบบคลาสซึ่งอ้างอิงถึงชนิดของอ็อบเจ็กเช่น MovieClip ก็เป็นคลาสชนิดหนึ่ง รวมถึง text field , button , strings , numbers ทุกอย่างล้วนเป็นคลาสทั้งสิ้น

คลาสโดยพื้นฐานจะประกอบด้วย properties (ข้อมูลและตัวแปร) และ behaviors (การกระทำ) สำหรับพรอพเพอร์ตี้ประกอบด้วยตัวแปรสำคัญที่เก็บข้อมูลต่างๆของคลาสและ behaviors คือฟังก์ชันของคลาสซึ่งมักจะอ้างอิงถึง โดยเรียกว่าเมธอด

2.4.1 คลาสพื้นฐาน

ตัวอย่าง 2.5

```
package {
    public class MyClass {
        public var myProperty:Number = 100;
        public function myMethod()
            { trace("I am here"); }
        }
}
```

ตัวอย่าง 2.5 เป็นโครงสร้างคลาสพื้นฐาน โดยเริ่มจาก package ซึ่งเป็นตัวรวมคลาสทั้งหมดเข้าด้วยกัน และทำการประกาศคลาสซึ่งสามารถใส่ access modifiers ได้ การประกาศคลาส public หมายความว่าคลาสนี้สามารถเข้าถึงได้จากโค้ดภายนอกคลาสภายในคลาสประกอบด้วยตัวแปรแบบ public 1 ตัว และฟังก์ชันแบบ public 1 ตัว การประกาศแบบ public ทำให้โค้ดภายนอกคลาสสามารถเข้าถึงตัวแปรและฟังก์ชันนี้ได้ ถ้าประกาศแบบ private จะป้องกันไม่ให้โค้ดจากภายนอกคลาสเข้าถึงตัวแปร และฟังก์ชันนี้ การประกาศแบบ internal จะให้คลาสที่อยู่ภายใน package เดียวกันเข้าถึงได้ และการประกาศแบบ protected หมายถึงพรอพเพอร์ตี้นั้นสามารถเห็นได้เฉพาะคลาสที่สืบทอด ต่อจากคลาสนี้เท่านั้น

2.4.2 Package

package เป็นการจัดการคลาสซึ่งเป็นการจัดการกลุ่ม โครงสร้างแบ่งหมวดหมู่ ของคลาส เช่นถ้าคลาสอยู่ใน folder com/friend/physics/ จะต้องเขียนคลาส package ดังตัวอย่าง 2.6

ตัวอย่าง 2.6

```
package com.friend.physics
        { public class Utils { } }
```

เมื่อมีหลายคลาสที่เก็บไว้ใน folder physics ก็จะได้กลุ่มคลาสที่ทำงานด้าน physics โดยอยู่ใน package เดียวกัน

2.4.3 อิมพอร์ต

การจะดึงคลาสมาใช้ได้จะต้องอิมพอร์ตโฟล์เดอร์ / คลาสที่จะใช้งานเข้ามาเช่นเดียวกับภาษาอื่นๆ ดังตัวอย่าง 2.7

ตัวอย่าง 2.7

```
        Import com.friend.physics.Utils;
```

เป็นการดึงคลาสข้างต้นเพื่อจะนำมาใช้งาน ใน AS3 ต้องอิมพอร์ตทุกคลาสที่จะนำมาใช้เมื่อคลาสนั้นอยู่ต่าง package

2.4.4 คอนสตรัคเตอร์(Constructors)

สามารถตั้งคอนสตรัคเตอร์ของคลาสได้และคลาสจะเรียกคอนสตรัคเตอร์ให้ทำงานเมื่อมีการสร้างอ็อบเจ็กจากคลาส ดังตัวอย่าง 2.8

ตัวอย่าง 2.8

```
package {
        public class MyClass {
                public function MyClass(arg:String) {
                        trace("constructed");
                        trace("you passed " + arg); } } }
```

เมื่อมีการสร้างอินสแทนซ์จะมีเรียกฟังก์ชันคอนสตรัคเตอร์ ดังตัวอย่าง 2.9

ตัวอย่าง 2.9

```
var myInstance:MyClass = new MyClass("hello");
```

เป็นการสร้างตัวแปร myInstance ซึ่งเป็นอ็อบเจ็กของคลาส MyClass และ pass argument String "hello" เข้าไป เมื่อเกิดการสร้างคลาสอ็อบเจ็กก็จะเรียกคอนสตรัคเตอร์และรับ argument เข้าไป ดังนั้น ผลลัพธ์ก็คือฟังก์ชัน MyClass จะทำการแสดง "constructed" และ "you pass hello" เป็น output ออกมา

2.4.5 Inheritance

เป็นการสืบทอดคุณสมบัติของคลาสหมายถึงคลาสที่ได้รับคุณสมบัติทั้งหมดและเมธอดของคลาสที่สืบทอด ยกเว้นพรอพเพอร์ตี้และเมธอดที่เป็น private และคลาสที่สืบทอดมาสามารถ override properties หรือเมธอดของ superclass นั้นได้ ดังตัวอย่าง 2.10

ตัวอย่าง 2.10

```
        package {
                public class MyBaseClass {
                 public function sayHello():void
                 {  trace("Hello from MyBaseClass"); }
                }
}
        package {
                public class MySubClass extends MyBaseClass {
                 public function sayGoodbye():void {
                trace("Goodbye from MySubClass"); }
                }
        }
```

ตัวอย่าง MySubClass จะสืบทอดคลาสจาก MyBaseClass โดยการใช้ extends เมื่อมีการสร้างอ็อบเจ็ก ดังตัวอย่าง 2.11

ตัวอย่าง 2.11

```
var base:MyBaseClass = new MyBaseClass();
base.sayHello();
```

คลาสหลักมีเมธอด sayHello ก็สามารถเรียกใช้งานได้ ดังตัวอย่าง 2.12

ตัวอย่าง 2.12

```
var sub:MySubClass = new MySubClass();
sub.sayHello();
sub.sayGoodbye();
```

เมื่อสร้าง subobject จาก subClass ที่สืบทอดออกมา ซึ่งประกาศเมธอด sayGoodbye ไว้ แต่เนื่องจาก subClass สืบทอดคลาสมา ทำให้สามารถเรียกใช้คลาสจากคลาสหลักได้ด้วยคือ sayHello();

และถ้าต้องการเขียนทับเมธอดเดิมของ superClassจะใช้การ override ฟังก์ชันได้ดัง ตัวอย่าง 2.13

ตัวอย่าง 2.13

```
package
{
        public class MySubClass extends MyBaseClass {
                override public function sayHello():void
                        { trace("Hola from MySubClass");  }
        public function sayGoodbye():void
                        { trace("Goodbye from MySubClass"); }
        }
}
```

ฟังก์ชัน sayHello ได้ถูก override ใหม่ซึ่งเมื่อมีการเรียกใช้เมธอดนี้ก็จะเรียกจากเมธอด ที่ overrideใหม่

2.4.6 MovieClip / Sprite subclass

เมื่อเขียนภาษาแอกชันสคิปท์สำหรับทุกๆอย่างที่ใส่โค้ดลงบนไทม์ไลน์ในแฟลชนั่นคุณกำลังใช้คลาสของ MovieClip

MovieClip class เป็น template สำหรับทุกพรอพเพอร์ตี้และเมธอดทุกตัวที่อยู่บน movieClip และ stage ซึ่งประกอบด้วยพรอพเพอร์ตี้ที่เหมือนกันคือ x , y position , scale และอื่นๆ ใน AS3 ได้เพิ่มคลาส Sprite ซึ่งเปรียบเหมือนกับคลาส MovieClip ที่ไม่มีไทม์ไลน์เมื่อมีการใช้โค้ดในการเขียนซึ่งไม่ได้ใช้งานในส่วนของไทม์ไลน์และเฟรมดังนั้น การใช้ Sprite จะทำให้โปรแกรมเบากว่าการใช้ MovieClip

ถ้าเขียนคลาสที่สืบทอดมาจาก MovieClip หรือ Sprite flash จะ inherit properties และเมธอดให้อัตโนมัติ ดังนั้นสามารถเพิ่ม คุณสมบัติพิเศษที่จะเขียนได้เลยโดยไม่ต้องมาสนใจ behavior ต่างๆของ MovieClip/Sprite

2.5 ทาร์เก็ตพาธ

ทาร์เก็ตพาธ (Target Path)นั้นถ้าแปรตรงๆก็คือ เส้นทางสู่เป้าหมาย ให้นึกถึงเวลานั่งรถไฟจำเป็นต้องนั่งไปทีละสถานีตามลำดับ ไม่สามารถลัดสถานีได้ แต่เส้นทางที่รถไฟวิ่งไปนั้น อาจจะมีเส้นทางแยกที่จะวิ่งไปสถานีอื่นๆที่แตกต่างกันได้

สำหรับในแฟลชนั้น เป้าหมายของก็คือ มูฟวี่คลิป ปุ่ม เทกซ์ฟิลด์ ตัวแปร และฟังก์ชันนั่นเอง

2.5.1 รูปแบบทาร์เก็ตพาธ

การที่จะสั่งให้อะไรทำงาน จะต้องกำหนดเป้าหมายมาก่อน ซึ่งการระบุถึงทาร์เก็ตหรือเป้าหมายนั้น แรกเริ่มเดิมทีในเวอร์ชันเก่าอย่างเช่น 4 และ 5 จะใช้คำว่าที่ชื่อว่า tellTarget เพื่อบอกว่าให้ทาร์เก็ตนั้นทำอะไรต่อไป ดังตัวอย่าง 2.14

ตัวอย่าง 2.14

```
tellTarget("myTarget")
{ play(); }
```

ซึ่งจากคำสั่งที่ผ่านมา จะเป็นการสั่งให้มูฟวี่คลิปชื่อ myTarget เล่นเฟรมภายในมูฟวี่คลิปนั้นๆ สำหรับการระบุไปยังมูฟวี่คลิปที่ซ้อนกันอยู่แต่ละระดับก็จะใช้เครื่องหมาย "/" เป็นตัวคั่น

เช่น tellTarget("myTarget/myTargetInside") เป็นต้น หรือกรณีที่ต้องการย้อนกลับไปยังระดับพาธที่อยู่สูงกว่า จะใช้การระบุ "../" เป็นการย้อนถอยหลังไปหนึ่งระดับชั้น

แต่ภายหลังได้มีวิธีการ ระบุทาร์เก็ตพาธใหม่โดยตัดคำสั่ง tellTarget ออกไปแล้วสามารถระบุชื่ออินสแทนซ์ได้เลย แลใช้เครื่องหมาย "." มาเป็นตัวคั่นในแต่ละชั้น เรียกการอ้างอิงแบบ dot syntax

อย่างกรณีที่มีมูฟวี่คลิปซ้อน กันดังตัวอย่าง 2.15 (แต่ละชื่อคือชื่อ instance name ที่ถูกกำหนดตรงพาเนลพรอพเพอร์ตี้ไม่ใช่ชื่อของซิมโบล์ในไลบราลี)

ตัวอย่าง 2.15

```
mc1
    - mc2
        - mc3
```

กรณีการเขียนโค้ดไว้ที่เฟรมของไทม์ไลน์หลัก ณ จุดเริ่มต้น การอ้างอิงไปยัง mc3 ซึ่งอยู่ระดับชั้นในสุดก็จะเป็นดังตัวอย่าง 2.16

ตัวอย่าง 2.16

```
mc1.mc2.mc3
```

ซึ่งจะเห็นว่าไม่ได้มีอะไรซับซ้อน ขอเพียงแค่ระบุตามลำดับชั้นของอินสแทนซ์ที่ซ้อนกันแต่เนื่องจากว่า สามารถเขียนภาษาแอกชันสคิปท์ ณ ระดับไทม์ไลน์ใดก็ได้ ฉะนั้น การอ้างอิงวัตถุ จะถูdแบ่งออกเป็น 2 รูปแบบคือ

2.5.1.1 Absolute path

โดย Abosolute path เป็นการอ้างอิงไทม์ไลน์ที่อยู่ระดับสูงที่สุดขณะแสดงผลงานนั้นๆ (สามารถเล่น swf ได้หลายตัวพร้อมๆกันภายในงานๆหนึ่ง ซึ่งจะได้ศึกษาต่อไป) ซึ่งระดับไทม์ไลน์สูงสุด จะเรียกว่า root โดยวิธีการอ้างอิง root จะใช้คำสั่ง _root

เมื่อ เริ่มต้นสร้างไฟล์ fla ขึ้นมา ไทม์ไลน์แรกสุดที่เห็นก็คือไทม์ไลน์ที่อยู่สูงที่สุดของงานชิ้นนั้น โดยวิธีการทดสอบพาธนั้นสามารถทดสอบด้วยคำสั่ง trace() ซึ่งใช้เพื่อแสดงผล ซึ่งถ้าเขียนคำสั่ง ตัวอย่าง 2.17

ตัวอย่าง 2.17

```
trace(_root)
```

ไว้ที่ระดับไทม์ไลน์ไหนก็ตาม ก็จะได้ผลลัพธ์เดียวกันคือ _level0 ซึ่งถือเป็นระดับ root (ศึกษาเรื่อง _level ได้ในภายหลัง)

การ ใช้ root นั้นจะมีประโยชน์เมื่อมีวัตถุที่ซ้อนกันอยู่หลายระดับชั้นมากๆ แล้วต้องการอ้างอิงสิ่งใดๆบนไทม์ไลน์ระดับที่อยู่สูงมาก ใช้ _root ถือว่าเป็นทางเลือกที่ดีในการอ้างอิงจากจุดเริ่มต้น ซึ่งเมื่ออ้างอิงไปยังระดับไทม์ไลน์สูงสุด สามารถไล่ระดับพาธย้อนกลับลงมาได้

ตัวอย่างเช่นการเขียนโค้ดไว้ ที่เฟรมของไทม์ไลน์ มูฟวี่คลิป mc3 ต้องการอ้างอิงไปยัง mc1 สามารถระบุถึง mc1 ด้วยการเขียนโค้ดดังตัวอย่าง 2.18

ตัวอย่าง 2.18

```
_root.mc1
```

โดยสามารถทดสอบพาธด้วยการเขียนดังตัวอย่าง 2.19

ตัวอย่าง 2.19

```
trace(_root.mc1);
```

ซึ่งจะแสดงผลเป็น _level0.mc1 โดยในกรณีที่ไม่สามารถอ้างอิงมูฟวี่คลิปดังกล่าวได้ หน้าต่าง output จะแสดงเป็น undefined

การเก็บตัวแปรทั้งหมดเอาไว้ที่ root ที่เดียว ก็ถือเป็นการจัดระเบียบของตัวแปรเพื่อการอ้างอิงและแก้ไขได้สะดวก แต่ก็ให้พิจารณาตามโครงสร้างของงานชิ้นนั้นๆด้วย

2.5.1.2 Relative path

relative ถ้าแปลจะหมายถึง เครือญาติ บุคคลใกล้ชิด สำหรับการเขียนโปรแกรมนั้น relative path จะเป็นการพาธไปยังเป้าหมายที่ต้องการ โดยอิงจากตำแหน่ง ณ จุดที่เริ่มต้นทำการอ้างอิง โดยการอ้างอิงจากจุดเริ่มต้นนั้น สามารถที่จะอ้างอิงไปในระดับที่ลึกกว่า หรือระดับที่สูงกว่าก็ได้ (ระบุพาธย้อนกลับหรือถอยหลัง)

การอ้างอิงแบบ relative นั้นจะใช้คำสั่งอยู่ 2 ตัวคือ

1) this ซึ่งหมายถึงตำแหน่งปัจจุบัน

2) _parent เป็นการอ้างอิงแบบย้อนกลับขึ้นไปหนึ่งระดับชั้น

โดยแต่ละตัวมีวิธีการใช้งานดังนี้

1) this เมื่อมีการประกาศ this ก็หมายถึงการเริ่มต้นอ้างอิงจากตำแหน่งปัจจุบัน กรณีที่มีการซ้อนมูฟวี่คลิปเอาไว้ดังตัวอย่าง 2.20

ตัวอย่าง 2.20

```
mc1
   - mc2
      - mc3
```

ซึ่งถ้าทำการเขียนโค้ดที่ไทม์ไลน์ของ mc1 โดยใช้ trace(this); ผลลัพธ์จะแสดงดังตัวอย่าง 2.21

ตัวอย่าง 2.21

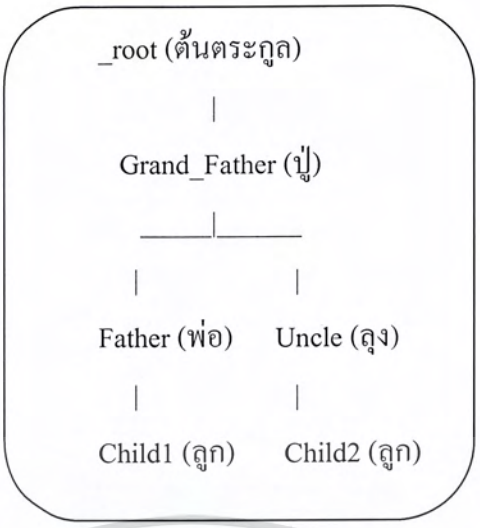
```
_level0.mc1;
```

และถ้าเขียนโค้ดที่ mc2 และ mc3 ด้วยคำสั่ง trace(this) ก็จะได้ _level0.mc1.mc2 และ _level0.mc1.mc2.mc3 ตามลำดับ

ซึ่งจะเห็นว่าพาธที่ได้จะนับเริ่มต้นจาก _root จนถึงตำแหน่งที่เขียนโค้ด

ข้อดีของการใช้ relative path มีประโยชน์ตรงที่ สามารถอ้างอิงมูฟวี่คลิปจากระดับชั้นที่ลึกมากๆได้เลย แทนที่จะอ้างอิงจาก _root ไล่ลำดับลงมา ซึ่งอาจจะมีหลายระดับชั้นที่ซ้อนกันอยู่

2) _parent หมายถึงระดับที่อยู่สูงกว่าตำแหน่งปัจจุบันย้อนไปหนึ่งระดับชั้น ซึ่งผังโครงสร้างการซ้อนกันแบบต้นไม้นี้ ถ้าจะมองเป็นแบบผังเครือญาติ ก็ย่อมได้ ถ้าวาดผังออกมาเป็นลำดับตระกูลและเครือญาติ ดังนี้



รูป 2.7 แผนผังลำดับเป็นต้นไม้

กรณีที่เขียนโค้ดที่ Child1 ถ้าต้องการจะอ้างอิงถึง Father โดยเริ่มต้นเขียนโค้ดที่ Child1 จะเห็นว่ามีการอ้างอิง this เพื่ออ้างอิงถึงตำแหน่งปัจจุบันก่อน ดังตัวอย่าง 2.22

ตัวอย่าง 2.22

```
trace(this._parent); //แสดงผลเป็น _level0.Grand_Father.Father
```

กรณีที่มีการอ้าง _parent ของ _parent ก็เท่ากับเป็นการย้อนขึ้นไปสองครั้งตั้งแต่ Child1 โดยให้เขียนโค้ดดังตัวอย่าง 2.23

ตัวอย่าง 2.23

```
trace(this._parent._parent); //ผลคือได้ _level0.Grand_Father
```

กรณีต้องการอ้างอิงไปยัง Child2 ไม่สามารถระบุได้ ดังตัวอย่าง 2.24

ตัวอย่าง 2.24

```
trace(this.Child2); //output ได้ undefined
```

ซึ่งถ้าดูจากผังระดับชั้นแล้ว ดูเหมือนว่า Child1 อยู่ระดับเดียวกับ Child2 แต่อันที่จริงแล้ว การจะมองเห็นกัน ต้องดูที่การซ้อนกันของมูฟวี่คลิปเป็นหลัก ดังนั้นการที่จะอ้างอิงวัตถุใดๆ จำเป็นต้องไล่ตามพาธในผังเท่านั้น ฉะนั้นการอ้างอิงที่ถูกต้องตามเส้นพาธจะเป็นดังตัวอย่าง 2.25

ตัวอย่าง 2.25

```
trace(this._parent._parent.Uncle.Child2);
//output แสดง  level0.Grand  Father.Uncle.Child2
```

ซึ่งเมื่อไรที่มีการอ้างอิง _parent หลายครั้ง ก่อนที่จะระบุวัตถุในตำแหน่งอื่นๆ อาจจะทำการ trace(this._parent._parent) ดูก่อนว่าย้อนพาธไปจนถึงระดับที่ต้องการหรือไม่ หรือถ้าเห็นว่าการระบุย้อนกลับไม่ต่างระดับชั้นกับการใช้ _root มากก็สามารถใช้ _root ได้เลย

2.5.2 การอ้างอิงพาธแบบ Dynamic (Dynamic Path)

นอกจากการอ้างอิงแบบปกติที่กล่าวมาแล้ว ยังสามารถที่จะอ้างอิงด้วยรูปแบบที่ยืดหยุ่นกว่าเดิมด้วยการอ้างอิงพาธ แบบ dynamic ซึ่งจะช่วยลดการอ้างอิงแบบซ้ำๆ ลงไปได้ โดยรูปแบบการอ้างอิงพาธแบบ dynamic จะเป็นดังนี้

กรณีอินสแทนซ์ ตัวแปรพรอพเพอร์ตี้และ dynamic text สามารถอ้างอิงได้เหมือนๆกันได้ดังตัวอย่าง 2.26

ตัวอย่าง 2.26

```
path[String]  หรือ  path[String + variable]
```

กรณีฟังก์ชันรวมถึงเมธอดสามารถอ้างอิงด้วยรูปแบบดังตัวอย่าง 2.27

ตัวอย่าง 2.27

```
path[String]() หรือ  path[String + variable]()
```

โดย path คือ พาธที่อยู่ก่อนหน้าเป้าหมายที่ต้องการอ้างอิง โดยระบุพาธตามปกติ แต่สำหรับหน้า [] ให้ละเว้นการระบุเครื่อง "."

String คือ ค่าสตริงใดๆ

variable คือตัวแปรใดๆก็ตาม (กรณีที่ variable มีการบวกค่ากับตัวเลขใดๆ ให้ใช้วงเล็บครอบเอาไว้ด้วยเช่น path[String + (variable+1)] มิฉะนั้นค่าหมายเลข 1 จะถูกแปลงให้เป็น String โดยอัตโนมัติ

กรณีอ้างอิงพาธขั้นต้นแบบ Absolute โดยต้องการอ้างอิงถึง mc3 จะระบุดังตัวอย่าง 2.28

ตัวอย่าง 2.28

```
_root.mc1.mc2["mc3"] หรือ _root.mc1.mc2["mc"+3];
```

สำหรับกรณีที่มีมูฟวี่คลิปหลายๆตัวเช่น mc1,mc2, mc3 วางอยู่ที่เดียวกันหมด การอ้างอิงถึงมูฟวี่คลิปแต่ละตัวจะเป็นการเสียเวลา ฉะนั้นมักนิยมที่จะนำคำสั่ง for ซึ่งใช้ในการวนลูปมาช่วย ดังตัวอย่าง 2.29

ตัวอย่าง 2.29

```
for (var i=1,i<=3;i++)  // สั่งให้วนลูป 3 รอบ โดยค่า i จะมีค่าตั้งแต่ 1 ถึง 3
    { trace(_root["mc" + i]); }
```

ผลลัพธ์ที่ได้จะเป็นดังตัวอย่าง 2.30

ตัวอย่าง 2.30

```
_level0.mc1
_level0.mc2
_level0.mc3
```

ซึ่งจะเห็นว่า นำตัวแปร i มาบวกเข้ากับ String และใช้ [] ครอบเท่านั้นเอง ก็จะช่วยลดระยะเวลาในการเขียนโค้ดได้มากขึ้น และยังทำให้โค้ดมีความกระชับ

2.5.2.1 ตัวอย่างอ้างอิงฟังก์ชันแบบ dynamic

สำหรับการใช้กับฟังก์ชัน ก็เพียงแค่เติม () ต่อท้าย [] ก็จะทำให้สามารถเรียกฟังก์ชันที่ต้องการได้แล้ว

ตัวอย่างการเรียกฟังก์ชันแบบ dynamic เช่น เขียนฟังก์ชันเอาไว้ฟังก์ชันหนึ่ง และต้องการให้เรียกฟังก์ชันที่ต้องการหลังจากทำงานฟังก์ชันนั้นจบ แต่บางครั้งฟังก์ชันที่จะเรียกหลังทำงานจบ อาจจะไม่ใช่ฟังก์ชันเดียวกันเสมอไป ฉะนั้นจะเขียนโค้ดดังตัวอย่าง 2.31

ตัวอย่าง 2.31

```
function  testDynamicFunction(path, callback, param) {
        //statement
        path[callback](param);
}
function myCallBack(param) {
        trace(param);
}
testDynamicFunction(this,"myCallBack", "myCallBack is called");
```

ซึ่งจะแสดงผล myCallBack is called ซึ่งการเรียกฟังก์ชัน testDynamicFunction จะส่งพาธเข้าไป โดยการระบุ this ก็คือตำแหน่งปัจจุบัน และส่งฟังก์ชันที่ต้องการเรียกกลับหลังจากจบการทำงาน ก็คือ myCallBack และมีการส่งพารามิเตอร์ให้กับฟังก์ชันอีกด้วย เมื่อ myCallBack ถูกเรียก จึงนำค่าพารามิเตอร์ดังกล่าวมาแสดงผล

2.5.3 การประกาศ ตัวแปร (variable) และ ฟังก์ชัน (function) ให้กับทาร์เก็ต(target)

สามารถทำการประกาศตัวแปร และฟังก์ชันให้กับทาร์เก็ตหรืออินสแทนซ์ที่ต้องการด้วยรูปแบบง่ายๆ หรือแบบ dynamic โดย variable คือชื่อตัวแปรและ value คือค่าที่ต้องการกำหนดให้ ดังตัวอย่าง 2.32

ตัวอย่าง 2.32

```
target.variable = value//simple    หรือ    target["variable"] = value//dynamic
```

ส่วนการประกาศฟังก์ชันจะมีรูปแบบดังตัวอย่าง 2.33

ตัวอย่าง 2.33

```
target.variable = function() {
    //statement }
                                    หรือ
function functionName = function() {
    //statement }
target.variable = functionName
```

หรือจะประกาศโดยใช้หลักการแบบ dynamic ก็ย่อมได้เช่นเดียวกันดังตัวอย่าง 2.34

ตัวอย่าง 2.34

```
target["functionName"] = function() {
    //statement }
                                    หรือ
target["functionName"+variable] = function() {
    //statement }
```

ส่วนวิธีการเรียกใช้นั้น จะเรียกใช้แบบอ้างอิงชื่อโดยตรง หรือแบบ dynamic ก็ย่อมได้เช่นเดียวกัน

สำหรับเรื่องนี้ ได้เรียนรู้การระบุพาธ และอ้างอิงไปถึงเป้าหมายที่ต้องการแล้ว ซึ่งถือว่าเป็นพื้นฐานและหัวใจสำคัญเบื้องต้นสำหรับแฟลชอย่างมาก กรณีที่ไม่สามารถอ้างอิงไปยังเป้าหมายที่ต้องการได้ ก็ไม่สามารถที่จะควบคุมสิ่งต่างๆให้เป็นไปตามที่ต้องการได้ ฉะนั้นควรทำความเข้าใจเรื่องนี้อย่างลึกซึ้งและสามารถอ้างอิงเป้าหมายต่างๆ จนเคยชิน ซึ่งจะช่วยให้การเขียนโปรแกรมง่ายยิ่งขึ้น

2.6 เมธอด(Method)

เมธอดคือวิธีการสำหรับกรณีนี้จะเรียกว่าการกระทำ หรือ การดำเนินการ แต่ถ้าแปลให้เข้าใจง่ายในเชิงโปรแกรมเชิงวัตถุเมธอดก็เปรียบเสมือนความสามารถที่วัตถุนั้นสามารถกระทำได้นั่นเอง ซึ่งอันที่จริงแล้วเมธอดนั้นก็คือฟังก์ชันที่ถูกกำหนดไว้ใน คลาสนั้นๆนั่นเอง โดยรูปแบบการใช้งานจะไม่ต่างอะไรกับฟังก์ชันมากนักโดยรูปแบบการใช้งานเป็น ดังตัวอย่าง 2.35

ตัวอย่าง 2.35

```
target.method();          หรือ
target.method(parameter1,parameter2,...,parameterN);
```

ซึ่งค่าพารามิเตอร์ที่จะให้กรอกนั้น ขึ้นอยู่กับว่าเมธอดนั้นถูกกำหนดให้สามารถส่งค่าอะไรได้บ้าง นอกจากนี้เมธอดเดียวกันไม่จำเป็นต้องส่งค่าพารามิเตอร์รูปแบบเดียวกันเสมอไป

2.6.1 การควบคุมไทม์ไลน์(Timeline) และ มูฟวี่คลิปไทม์ไลน์(Movie Clip Timeline)

โดยปกติแล้ว มักจะได้เรียนวิธีการควบคุมไทม์ไลน์หรือตำแหน่งของ play head เป็นพื้นฐานก่อนเสมอในเบื้องต้น โดยจะควบคุมการเล่น การหยุด หรือการกระโดดไปยังเฟรมอื่น โดยคำสั่งที่ใช้ควบคุมทั้งหมดจะมีดังตัวอย่าง 2.36

ตัวอย่าง 2.36

play()	สำหรับเล่น
stop()	สำหรับหยุด
nextFrame()	ไปข้างหน้า 1 เฟรมจากตำแหน่งเฟรมปัจจุบัน
prevFrame()	ถอยหลัง 1 เฟรมจากตำแหน่งเฟรมปัจจุบัน
gotoAndPlay(frame)	กระโดดไปยังเฟรมที่ต้องการและทำการเล่น
gotoAndPlay(label)	กระโดดไปยังเฟรมที่มีการประกาศชื่อลาเบลที่ระบุ
gotoAndStop(frame)	กระโดดไปยังเฟรมที่ต้องการและทำการหยุด ณ เฟรมนั้น
gotoAndStop(label)	กระโดดไปยังเฟรมที่มีการประกาศชื่อลาเบลที่ระบุ และหยุด
gotoAndPlay(Scene,frame)	กระโดดไปยังซีนที่ต้องการ และกำหนดค่าเฟรมที่ต้องการในซีนนั้นๆ และกำหนดให้เล่นต่อจากเฟรมดังกล่าว
gotoAndStop(Scene,frame)	กระโดดไปยังซีนที่ต้องการ และกำหนดค่าเฟรมที่ต้องการในซีนนั้นๆ และกำหนดให้หยุดที่เฟรมดังกล่าว

กรณีมีการกำหนดให้ข้ามเฟรมโดยระบุค่าเป็นชื่อซีน ถือเป็นกรณีพิเศษที่ห้ามระบุพาธใดๆ นำหน้าทั้งสิ้น ซึ่งถ้ามีการกำหนดพาธนำหน้าคำสั่ง จะทำให้คำสั่งไม่สามารถทำงานได้ ซึ่งการกระโดดแบบซีนนั้น จะใช้ได้เมื่อเขียนโค้ดเอาไว้ในไทม์ไลน์ใดๆ หรือกำหนดภายในอีเวนต์ของอินสแทนซ์ประเภท Button เท่านั้น (Button มีคุณสมบัติต่างจาก MovieClip อยู่อย่างนึงก็คือ ถ้าไม่ระบุพาธ พาธที่เป็นค่า default จะเปรียบได้กับการระบุ _parent ซึ่งจะทำให้ย้อนขึ้นไปหนึ่งระดับชั้นแทนที่จะหมายถึงตัว Button นั้นๆ) ให้ทำความเข้าใจในหัวข้ออีเวนต์เสริมเพิ่มเติมในภายหลัง

กรณีที่ต้อง การควบคุมไทม์ไลน์ของ MovieClip อาทิเช่น มี MovieClip ตั้งชื่อ instance name ไว้ว่า my_mc ก็สามารถกำหนดได้ดังต่อไปนี้ (เขียนโค้ดบนไทม์ไลน์ในระดับที่ my_mc วางอยู่) จะเห็นว่าวิธีการนั้นง่ายๆก็คือการกำหนดเป้าหมายหรือระบุทาร์เก็ตพาธก่อนแล้วจึงทำการเรียกเมธอดที่ต้องการ ดังตัวอย่าง 2.37

ตัวอย่าง 2.37

```
my_mc.play();
```

2.6.2 ความสัมพันธ์ระหว่างเมธอดและฟังก์ชัน

จากได้บอกไปแล้วในตอนต้นว่าเมธอดก็คือฟังก์ชันที่อยู่ภายในคลาสฉะนั้นจะเห็นว่ารูปแบบการใช้งานเมธอด จึงไม่ต่างอะไรกับการใช้งานฟังก์ชัน

และถ้าสังเกตุให้ดี การเรียกใช้งานเมธอด นั้นก็เหมือนการเรียกฟังก์ชันซึ่งอยู่ใน MovieClip ฉะนั้นรูปแบบที่เรียกจึงออกมาแบบ mc.function() นั่นเอง ซึ่ง MovieClip ก็ถือว่าเป็นคลาสประเภทนึง

อาจจะพูดได้ว่าเมธอดก็คือ built in function หรือฟังก์ชัน ภายในที่ได้ถูกสร้างเอาไว้แล้วภายในคลาสนั้น จึงสามารถเรียกใช้งานได้ทันที

2.6.1.1 คำสั่งที่เป็นทั้งเมธอด และฟังก์ชัน

บางครั้งบางคลาสนั้นก็สามารถที่จะใช้ในรูปแบบฟังก์ชันได้ด้วย สาเหตุเนื่องจากแต่เดิมทีนั้น คำสั่งนั้นถูกใช้งานในรูปแบบฟังก์ชันมาก่อน ก่อนที่จะมีการใช้งานทาร์เก็ตพาธ ตัวอย่างเช่นคำสั่ง duplicateMovieClip() ซึ่งใช้ในการสร้าง MovieClip ขึ้นมาใหม่จากต้นแบบทาร์เก็ตที่กำหนด โดยเมื่อพิมพ์คำสั่ง duplicateMovieClip (โปรแกรมจะแสดง short cut เพื่อแสดงค่าพารามิเตอร์ที่สามารถกำหนดได้ดังตัวอย่าง 2.38

ตัวอย่าง 2.38

```
duplicateMovieClip(target,newname="",depth);
```

ซึ่งจะเห็นว่ามีการให้กำหนดเป้าหมายหรือ MovieClip ต้นแบบที่จะนำมาสร้างใหม่ชื่อใหม่ของ MovieClip ที่ถูกสร้างและสุดท้ายคือระดับความลึกของ MovieClip ที่สร้าง

ซึ่งรูปแบบดังกล่าวเป็นรูปแบบการใช้งานแบบฟังก์ชันกรณีที่พิมพ์คำสั่งในรูปแบบเมธอด เช่น my_mc.duplicateMovieClip (โปรแกรมจะแสดง short cut เพื่อแสดงค่าพารามิเตอร์ที่สามารถกำหนดได้ดังตัวอย่าง 2.39

ตัวอย่าง 2.39

```
my_mc.duplicateMovieClip(newname="",depth, [initObject])
```

ซึ่งการกำหนดค่าทาร์เก็ตจะหายไป แต่จะมีให้กำหนด initObject มาแทน ซึ่งเป็นวัตถุที่เก็บคุณสมบัติต่างๆเอาไว้และต้องการถ่ายทอดคุณสมบัติดัง กล่าวให้กับ MovieClip ที่ถูกสร้างขึ้นมาใหม่

ในการกำหนดค่าพารามิเตอร์ ซึ่งค่าใดที่มี [] ครอบอยู่ จะหมายถึงทางเลือกที่คุณสามารถส่งค่าหรือไม่ก็ได้ (โดยส่วนใหญ่ เอกสารคู่มือหรือ help สำหรับแต่ละภาษาจะใช้สัญลักษณ์นี้เช่นเดียวกัน)

2.6.2 การใช้เมธอด ของคลาสหนึ่งเพื่อควบคุมมูฟวี่คลิปและคลาสที่ทำงานด้วยตัวเอง

ในการสร้างอ็อบเจ็กจากคลาสใดๆนั้น การสร้างคลาสนั้นจะแบ่งเป็นคลาสอยู่ 2 ประเภท

1) คลาสที่สร้างอ็อบเจ็กขึ้นมาเองโดยไม่อิงกับ มูฟวี่คลิป เช่น Date, เมาส์, คีย์บอร์ด และอื่นๆ

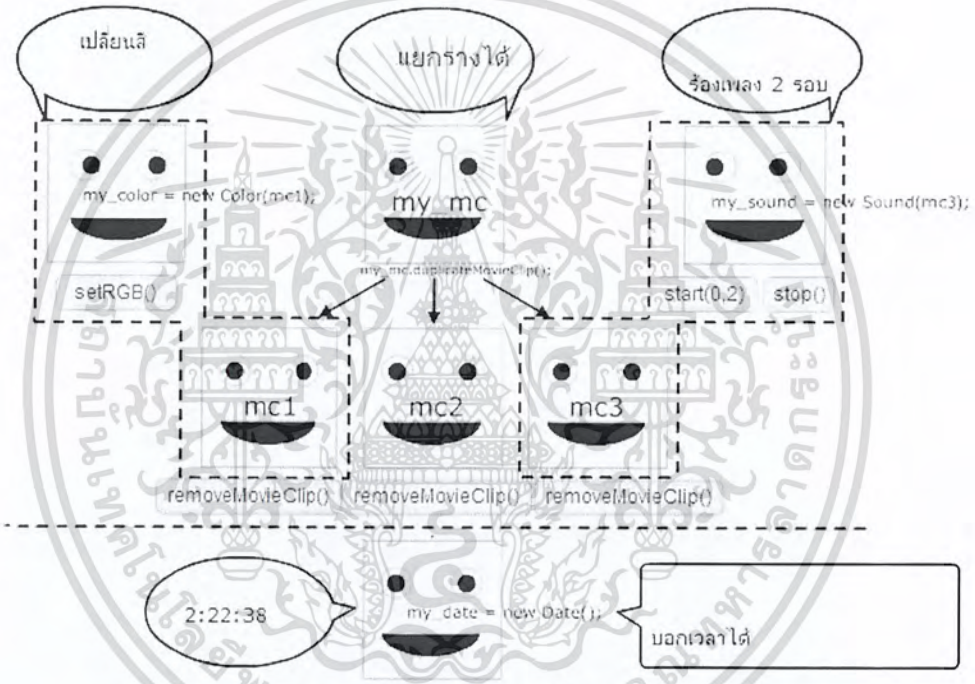
2) คลาสที่สร้างอ็อบเจ็กขึ้นมาโดยอิงกับมูฟวี่คลิปเช่น Sound, Color และอื่นๆ โดยการสร้างคลาสประเภทที่ 2 นั้นจะต้องทำการอ้างอิง มูฟวี่คลิป ตอนสร้าง ตัวอย่าง เช่น คลาส Sound หรือ Color เป็นต้น โดยตัวอย่างการสร้างจะเป็นดังตัวอย่าง 2.40

ตัวอย่าง 2.40

```
my_sound = new Sound(my_mc);
my_color = new Color(my_mc);
```

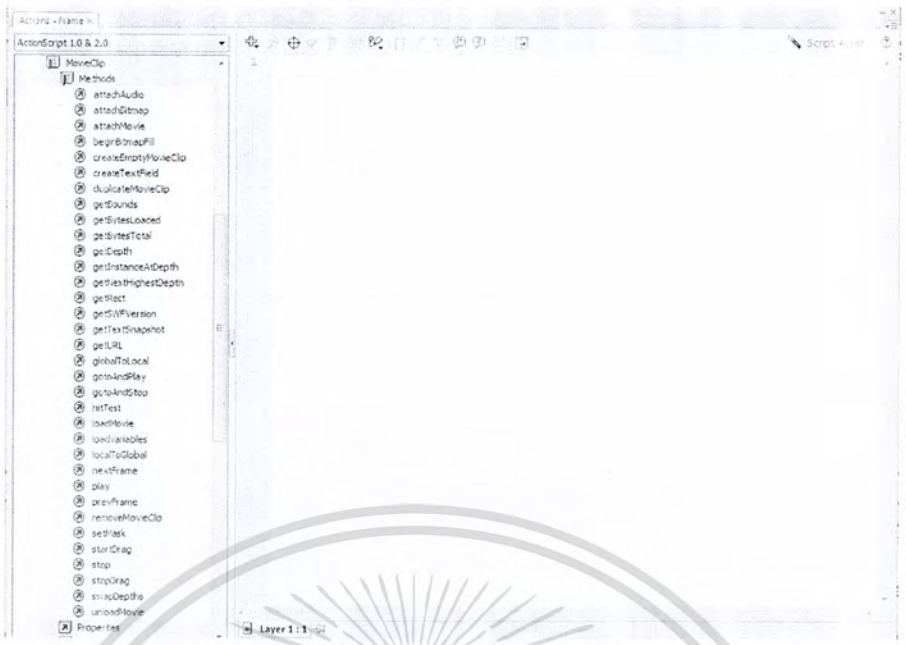
สำหรับ Sound นั้นกรณีไม่ระบุทาร์เก็ตลงไป ก็จะทำการอ้างอิงไทม์ไลน์ที่เขียนโค้ดนั้นๆ โดยอัตโนมัติ ส่วน Color นั้น สาเหตุที่จำเป็นต้องระบุทาร์เก็ตก็เพื่อให้รู้ว่าวัตถุใดที่ต้องการทำการเปลี่ยนสีนั่นเอง

จากด้านล่างเป็นตัวอย่างของการเรียกใช้เมธอดของแต่ละคลาสโดยมูฟวี่คลิปนั้นสามารถที่จะใช้เมธอดชื่อ duplicateMovieClip() ในการสร้างวัตถุนั้นๆขึ้นมาใหม่ และสามารถลบทิ้งได้ด้วยคำสั่ง removeMovieClip() เป็นต้น นอกจากนี้ เมื่อทาร์เก็ตที่ถูกอ้างอิงหายไปอ็อบเจ็กที่สร้างขึ้นจากคลาสนั้นก็จะไม่สามารถทำงานได้อีกต่อไป (ภายในตัวอย่างไฟล์ fla จะขอละเว้นการใช้คำสั่ง duplicateMovieClip() ออกไปเพื่อลดความซับซ้อนของตัวอย่างการเขียนโค้ด)



รูป 2.8 อธิบายการทำงานความสัมพันธ์ของคลาสและวัตถุต่างๆ

ในเรื่องนี้ได้เรียนรู้การใช้งานเมธอดเรียบร้อยแล้ว ซึ่งวิธีการนั้นก็ง่ายๆอย่างที่ได้ยกตัวอย่างไป ซึ่งคำสั่งเมธอดจะแตกต่างกันไปสำหรับคลาสแต่ละประเภท ซึ่งจะได้เรียนรู้ในการใช้งานคลาสแต่ละประเภทว่ามีเมธอดอะไรที่ใช้บ่อยและน่าสนใจบ้างแยกย่อยในหมวดการเรียนรู้และใช้งานคลาสแต่ละประเภท ซึ่ง ทีเมธอดควรศึกษาเป็นอันดับต้นๆก็คือเมธอดของซิมโบล์อย่างเช่นมูฟวี่คลิปและ Button เนื่องจากถือว่าเป็นคลาสที่ใช้งานค่อนข้างบ่อย เนื่องจากเกี่ยวข้องกับการแสดงผลเป็นหลัก



รูป 2.9 ตัวอย่างเมธอดของ MovieClip

2.7 พรอพเพอร์ตี้(Properties)

สำหรับ properties นั้น หรือ property แปลได้ว่า คุณสมบัติ ซึ่งคุณสมบัติเป็นสิ่งที่แสดงลักษณะอย่างใดอย่างหนึ่งของวัตถุหรือคลาสใดๆ อย่างเช่นเรื่องของขนาด ความสูง ความกว้าง หรือแม้แต่เสียง (Sound) ก็มีเรื่องของความดัง (volume) เป็นต้น (แต่บางครั้งอาจจะถูกกำหนดค่าผ่านเมธอดก็ได้)

ถ้ามองในแง่ของการเขียนโปรแกรมเชิงวัตถุพรอพเพอร์ตี้ นั้นก็เทียบได้กับตัวแปร (variable) ที่ถูกกำหนดภายในคลาสนั้นๆ ซึ่งหนังสือบางเล่มอาจจะเรียกว่า instance variable (อินสแทนซ์หรืออ็อบเจ็กนั้นถูกสร้างขึ้นจากคลาส) ดังนั้นรูปแบบการใช้งานพรอพเพอร์ตี้ จึงไม่ต่างอะไรกับการอ้างถึงตัวแปรที่อยู่ภายในเป้าหมาย (target) ที่ต้องการ (ถ้านึกไม่ออก ให้คิดถึงตัวแปรที่ประกาศอยู่ใน MovieClip)

2.7.1 รูปแบบการใช้งานพรอพเพอร์ตี้

รูปแบบการใช้งานจะแบ่งเป็น 2 กรณีดังต่อไปนี้

2.7.1.1 พรอพเพอร์ตี้แบบที่อ่านค่าได้อย่างเดียว

ซึ่งจะสามารถเรียกใช้งานเพื่อดูค่าได้เท่านั้น ไม่สามารถกำหนดค่าอะไรให้ได้ โดยจะมีรูปแบบดังตัวอย่าง 2.41

ตัวอย่าง 2.41

```
target.property
```

โดยทำการระบุทาร์เก็ตที่ต้องการตามหลักการระบุทาร์เก็ตพาธก่อน หลังจากนั้นจึงทำการระบุพรอพเพอร์ตี้ของคลาสนั้นๆ

2.7.1.2 พรอพเพอร์ตี้แบบที่อ่านค่าได้และกำหนดค่าได้

สำหรับการอ่านค่านั้นจะมีรูปแบบเดียวกันกับข้อ 1 ส่วนวิธีการกำหนดค่า จะใช้รูปแบบดังตัวอย่าง 2.42

ตัวอย่าง 2.42

```
target.property = value
```

ซึ่งวิธีการกำหนดค่า เพียงแค่ใช้เครื่องหมายเท่ากับ และทางด้านขวาก็คือค่าที่จะกำหนด ซึ่งต้องมีชนิดข้อมูลตรงกับที่พรอพเพอร์ตี้นั้นต้องการ ซึ่งจะเห็นว่ารูปแบบการกำหนดค่าเหมือนกับการกำหนดค่าให้กับตัวแปรนั่นเอง

2.7.2 พรอพเพอร์ตี้ยุคแรกกับยุคหลัง

ในช่วงแรกของภาษาแอกชันสคิปท์นั้น รูปแบบคำสั่งพรอพเพอร์ตี้นั้นจะใช้ "_" นำข้างหน้า ตัวอย่างเช่นพรอพเพอร์ตี้หลายๆตัวของ MovieClip จะเป็นดังนี้

_x อ่านค่าหรือกำหนดตำแหน่งในแนวแกน x บน Stage

_y อ่านค่ากำหนดตำแหน่งในแนวแกน y บน Stage

_width อ่านค่าหรือกำหนดความกว้างให้กับ MovieClip หน่วยเป็นพิกเซล

_height อ่านค่าหรือกำหนดความสูงให้กับ MovieClip หน่วยเป็นพิกเซล

_xscale อ่านค่าหรือกำหนดความกว้างให้กับ MovieClip หน่วยเป็นเปอร์เซ็นต์

_yscale อ่านค่าหรือกำหนดความสูงให้กับ MovieClip หน่วยเป็นเปอร์เซ็นต์

จากด้านบน เป็นตัวอย่างพรอพเพอร์ตี้บางส่วนของ MovieClip เท่านั้น ซึ่งจะเห็นว่าจะมี _ นำหน้าอยู่ แต่ตอนหลังพรอพเพอร์ตี้หลายๆตัวได้ถูกตัด _ ออกไป (ส่วนใหญ่จะมีเหลือแค่ของ MovieClip ที่ยังคงไว้ตามเดิมเพื่อยังคงสนับสนุนไฟล์งานที่พัฒนาเอาไว้ก่อนหน้านั้น) โดยเฉพาะ AS3 จะทำการตัดออกไปเลย จาก _x ก็จะเป็น x เฉยๆ ฉะนั้นบางทีไม่ควรกำหนดตัวแปรให้กับมูฟวี่คลิปด้วยชื่อที่ซ้ำกับพรอพเพอร์ตี้ถึงแม้จะไม่มีข้อห้ามก็ตาม

2.6.3 พรอพเพอร์ตี้ที่ถูกกำหนดค่าและดึงค่าด้วยเมธอดแทน

บางครั้งการกำหนดค่าพรอพเพอร์ตี้หรือ ดึงค่าพรอพเพอร์ตี้นั้นไม่ได้อยู่ในรูปแบบของการกำหนดค่าแบบตัวแปรตรงๆเสมอไป แต่อาจจะกำหนดค่าผ่านทางเมธอดอีกทีก็ได้ ซึ่งการ

เลือกใช้เมธอดเพื่อกำหนดพรอพเพอร์ตี้นั้นจะมีข้อดีกว่าตรงที่สามารถกำหนดเงื่อนไขให้กับค่าที่จะกำหนดให้กับพรอพเพอร์ตี้นั้นๆได้ ดังตัวอย่าง 2.43

ตัวอย่าง 2.43

```
setProperties()
getProperties()
```

ซึ่งเมธอดที่ขึ้นต้นด้วย set ก็คือการกำหนดค่า และเมธอดที่ขึ้นต้นด้วย get ก็คือสำหรับดึงค่า ซึ่งในตอนนี้ยังไม่ได้เริ่มต้นเขียนโปรแกรมในรูปแบบของโปรแกรมเชิงวัตถุ(ซึ่งจะได้ศึกษาในภายหลัง ตอนนี้เพียงแค่รู้คร่าวๆ) ฉะนั้นจะยกตัวอย่างในรูปแบบการเขียนฟังก์ชันภายในมูฟวี่คลิป โดยสมมุติว่า บนไทม์ไลน์มูฟวี่คลิปชื่อ my_mc มีการเขียนฟังก์ชันดังตัวอย่าง 2.44

ตัวอย่าง 2.44

```
function setSize(w,h)
        { this._width = w; //กำหนดความกว้าง
          this._height = h; //กำหนดความสูง }
function getWidth()
        { return this._width; //คืนค่าความกว้าง }
function getHeight()
        { return this._height; //คืนค่าความสูง }
```

เมื่อเรียกใช้ ก็จะเรียกใช้ในรูปแบบเมธอดตามปกติ ตัวอย่าง 2.45

ตัวอย่าง 2.45

```
my_mc.setSize(100,50);
myWidth = my_mc.getWidth();
myHeight = my_mc.getHeight();
trace(myWidth); //100
trace(myHeight); //50
```

สำหรับการกำหนดหรือดึงค่าพรอพเพอร์ตี้ในรูปแบบเมธอดนั้นจะมีการใช้งานอย่างมาก ในตอนที่มีการสร้างคลาสขึ้นมาเอง

ตัวอย่างเมธอดที่ดึงและกำหนดค่าพรอพเพอร์ตี้ของคลาส ที่มีอยู่แล้ว อาทิเช่น

1) Sound - จะมี setVolume(), getVolume()

2) Color - จะมี setRGB(), getRGB()

ซึ่งจะกำหนดให้การกำหนดพรอพเพอร์ตี้นั้นถูกกระทำผ่านตัวแปรโดยตรงหรือผ่านคลาสก็ขึ้นอยู่กับการดีไซน์คลาสของแต่ละคน

สำหรับในเรื่องนี้ได้เรียนรู้การใช้งานพรอพเพอร์ตี้ในการกำหนด และดึงค่ามาใช้เรียบร้อยแล้ว จะเห็นว่ารูปแบบการใช้งานนั้นไม่ได้ต่างอะไรกับการใช้งานตัวแปรเลยแม้แต่น้อย

2.8 อีเวนต์(Events)

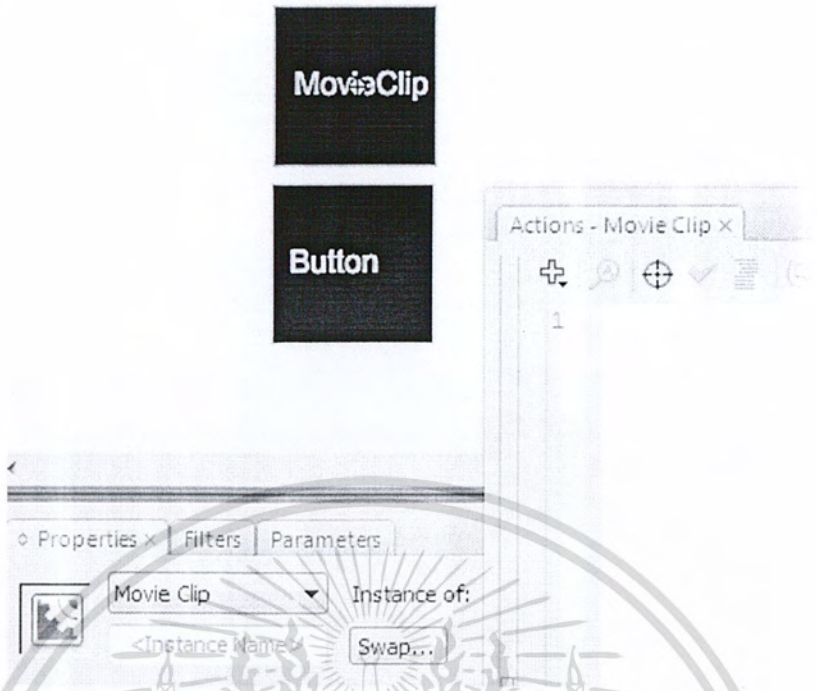
อีเวนต์ เป็นตัวที่กำหนดการตอบสนองสิ่งต่างๆให้เกิดขึ้นเมื่ออีเวนต์นั้นๆเกิดขึ้น เช่น กำหนดอีเวนต์ให้กับปุ่มๆหนึ่ง ซึ่งอาจทำการกำหนดเอาไว้ว่า ถ้าโดนกด ก็จะทำการเปลี่ยนเฟรม เป็นต้น ซึ่งคลาสแต่ละประเภทจะมีรูปแบบอีเวนต์หลายๆแบบรองรับ

จากหัวข้อ เริ่มต้นเขียนภาษาแอกชันสคิปท์ได้มีการอธิบายไปแล้วในหัวข้อว่า เขียน ภาษาแอกชันสคิปท์ที่ได้ไหนบ้าง ซึ่งสามารถเขียนได้ทั้งบนตัวอินสแทนซ์และ บนไทม์ไลน์ ของ MovieClip ซึ่งการเขียนบนตัวอินสแทนซ์จำเป็นจะต้องเขียนภายใต้อีเวนต์เท่านั้น อันที่จริงแล้ว ถ้าหากย้อนประวัติกลับไปที่ flash 5 เดิมทีสามารถกำหนดอีเวนต์บนตัวอินสแทนซ์ได้เพียงเท่านั้นซึ่งยังไม่สามารถกำหนดบนไทม์ไลน์ได้ นอกจากนี้อีเวนต์ของซิมโบล์ประเภท Button กับ MovieClip นั้นคำสั่งจะไม่ซ้ำซ้อนกัน ฉะนั้นการกดปุ่มเพื่อให้ทำคำสั่งใดๆ จำเป็นต้องใช้ซิมโบล์ประเภท Button เป็นหลัก แต่พอมาถึงในเวอร์ชัน flash mx ทางผู้พัฒนาได้มีการปรับปรุงโดยที่สามารถใช้ MovieClip แทนปุ่มได้อีกด้วย เนื่องจากได้เพิ่มอีเวนต์ของ Button ให้กับ MovieClip เรียบร้อยแล้ว

2.8.1 รูปแบบการกำหนดอีเวนต์ให้กับอินสแทนซ์ใดๆ

2.8.1.1 การกำหนดอีเวนต์บนตัวอินสแทนซ์

การกำหนดอีเวนต์บนตัวอินสแทนซ์นั้นสามารถทำได้โดยการคลิกที่อินสแทนซ์ที่ต้องการแล้วทำการเปิดแอกชันพาเนลขึ้นมา ซึ่งจากภาพตัวอย่างจะเห็นว่า MovieClip สี่เหลี่ยมนั้นโดนคลิกเลือกอยู่ (ขึ้นกรอบสีน้ำเงิน) และให้สังเกตชื่อตรง Actions Panel จะบอกว่ากำลังเขียนโค้ดอยู่ที่ไหน



รูป 2.10 การกำหนดอีเวนต์บนตัวอินสแทนซ์

สำหรับการประกาศอีเวนต์บนตัวอินสแทนซ์จะมีรูปแบบดังต่อไปนี้

1) กรณีอินสแทนซ์เป็น MovieClip จะประกาศดังตัวอย่าง 2.46

ตัวอย่าง 2.46

```
onClipEvent(event)
{ //statement }
```

โดยอีเวนต์คือชื่ออีเวนต์ต่างๆที่ MovieClip รองรับ

2) กรณีอินสแทนซ์เป็น Button จะประกาศดังตัวอย่าง 2.47

ตัวอย่าง 2.47

```
on(event)
{//statement }
```

โดยอีเวนต์คือชื่ออีเวนต์ต่างๆที่ Button รองรับ

การเขียนโค้ดบนตัวอินสแทนซ์นั้นจำเป็นต้องเขียนภายใต้อีเวนต์ เสมอ มิฉะนั้นโปรแกรมจะทำการเตือน error ซึ่งมักจะเกิดขึ้นบ่อยครั้งเมื่อเข้าใจว่ากำลังเขียนโค้ดอยู่บนไทม์ไลน์ในขณะที่ตำแหน่งจริงๆของโค้ดนั้นอาจจะอยู่บนอินสแทนซ์ตัวใดตัวหนึ่ง

2.8.1.2 การกำหนดอีเวนต์บนไทม์ไลน์

หลังจาก flash mx (flash 6) ออกมา ทำให้สามารถประกาศอีเวนต์ต่างๆให้กับอินสแทนซ์ได้สะดวกมากขึ้น โดยสามารถที่จะกำหนดอีเวนต์โดยการเขียนโค้ดบนไทม์ไลน์ ณ ตำแหน่งใดๆก็ได้ โดยอาศัยหลักการระบุทาร์เก็ตพาธเข้ามาช่วยอ้างอิงอินสแทนซ์ที่ต้องการ ซึ่งรูปแบบการประกาศอีเวนต์บนไทม์ไลน์จะเป็นดังตัวอย่าง 2.48

ตัวอย่าง 2.48

```
target.onEvent = function()
{  //statement }
```

หรือ การประกาศฟังก์ชันใดๆไว้แล้วนำฟังก์ชันดังกล่าวมาผูกหรือเชื่อมต่อ (handle) เข้ากับอีเวนต์นั้นๆ ดังตัวอย่าง 2.49

ตัวอย่าง 2.49

```
function functionName() {
    //statement }
target.onEvent = functionName
        หรือ
functionName = function() {
    //statement }
target.onEvent = functionName
```

โดย อีเวนต์ คือชื่ออีเวนต์ต่างๆที่ทาร์เก็ตนั้นๆ รองรับ (ให้ขึ้นต้นด้วยตัวอักษรใหญ่เป็นตัวแรก ซึ่งจะเป็นชื่อเดียวกับการประกาศอีเวนต์บนตัวอินสแทนซ์) และ functionName คือชื่อฟังก์ชันใดๆที่ได้ประกาศไว้ และต้องการให้ทำงานเมื่อเกิดอีเวนต์ขึ้น

2.8.2 อีเวนต์พื้นฐานของ Button และ MovieClip ที่ควรรู้

สำหรับอีเวนต์พื้นฐานของทั้ง Button และ MovieClip นั้นขอแสดงในรูปแบบการประกาศอีเวนต์บนไทม์ไลน์เพื่อให้ตรงกับที่แสดงผลในโปรแกรม ซึ่งจะขอยกตัวอย่างเมื่อนำไปใช้ในรูปแบบที่เขียนบนอินสแทนซ์เล็กน้อย เช่น คำสั่ง onPress ถ้าเขียนไว้บนตัวอินสแทนซ์จะใช้คำที่อยู่หลังคำว่า on และเป็นตัวอักษรเล็กขึ้นต้น โดยจะได้ดังตัวอย่าง 2.50

ตัวอย่าง 2.50

```
on(press) { }
```

บางคำสั่งอาจจะไม่รองรับในรูปแบบ on(event) เสมอไป เนื่องจากเป็นคำสั่งที่มาทีหลังและยึดรูปแบบการเขียนบนไทม์ไลน์เป็นหลัก

2.8.3 การอ้างอิง this ภายใต้อีเวนต์หรือฟังก์ชันที่ผูกอยู่กับอีเวนต์

โดยปกติแล้ว เมื่อมีการอ้างอิงถึง this ภายใต้ฟังก์ชันทั่วไป จะหมายถึงการอ้างถึงไทม์ไลน์ที่ฟังก์ชันนั้นโดนประกาศไว้ แต่กรณีที่มีการอ้างอิง this ภายใต้อีเวนต์ฟังก์ชันให้จำเอาไว้เสมอว่า this นั้นจะหมายถึงวัตถุที่ทำการประกาศอีเวนต์เอาไว้ดังตัวอย่าง 2.51

ตัวอย่าง 2.51

```
my_mc.onPress = function() {
    trace(this); // _level0.my_mc  โดยที่ my_mc อยู่บน _root หรือ Main Timeline }
```

หรือกรณีที่ฟังก์ชันนั้นถูกนำมาผูกกับอีเวนต์ก็จะได้ผลเช่นเดียวกัน ดังตัวอย่าง 2.52

ตัวอย่าง 2.52

```
function myEventHandle() {   //หรือ myEventHandle = function() {
    trace(this);
// _level0.my_mc  โดยที่ my_mc อยู่บน _root หรือ Main Timeline }
my_mc.onPress = myEventHandle;
```

ซึ่งการใช้ this เพื่ออ้างอิง จะช่วยให้อ้างอิงวัตถุนั้นๆได้ในทันที รวมถึงวัตถุที่อยู่ในระดับเดียวกัน โดยใช้ this._parent ช่วย

2.8.4 การเรียกใช้อีเวนต์ที่กำหนดไว้ในรูปแบบฟังก์ชัน

เมื่อทำการประกาศอีเวนต์ให้กับอ็อบเจ็กหรืออินสแทนซ์ผ่านทางไทม์ไลน์ นั้น ซึ่งถ้าดูรูปแบบการประกาศซึ่งอยู่ในลักษณะดังตัวอย่าง 2.53

ตัวอย่าง 2.53

```
my_btn.onPress = function()
{ trace("press"); }
```

จะเห็นได้ว่า รูปแบบของอีเวนต์ที่โดนประกาศจะไม่ได้ต่างอะไรกับการประกาศฟังก์ชันให้กับ มูฟวี่คลิปสักเท่าไร ดังนั้นสามารถที่จะทำการเรียกอีเวนต์ที่ถูกประกาศในรูปแบบของฟังก์ชัน ดังตัวอย่าง 2.54

ตัวอย่าง 2.54

```
my_btn.onPress = function()
{ trace("press"); }
my_btn.onPress();
//output ได้ press
```

นอกจากนี้ ยังสามารถกำหนดตัวแปรพารามิเตอร์ได้อีกด้วย ดังตัวอย่าง 2.55

ตัวอย่าง 2.55

```
my_btn.onPress = function(param)
{ trace(param); }
my_btn.onPress("press");
//output ได้ press
```

ซึ่งการเรียกใช้งานในรูปแบบฟังก์ชันนี้ จะมีประโยชน์ เช่น บางครั้งทำการประกาศอีเวนต์ให้กับปุ่มหลายๆด้วยการใช้คำสั่งวนลูปและอ้าง อิงปุ่มแต่ละปุ่มแบบ dynamic ซึ่งบางทีต้องการจะให้ปุ่มแรกนั้นโดนกดเอาไว้เรียบร้อยแล้ว ฉะนั้นสามารถเช็คค่าตัวแปรว่าเป็นค่าแรกของคำสั่งวนลูปหรือไม่ ซึ่งถ้าใช่ ก็สามารถที่จะกำหนดให้มีการกดปุ่มนั้นเกิดขึ้น ดังตัวอย่าง 2.56

ตัวอย่าง 2.56

```
//อ้างอิง btn1, btn2, btn3
for (var i=1;i<=3;i++)
{
    var btn = _root["btn"+i];
    btn.onPress = function() {
        trace(this); }
    if (i==1) {//ถ้าค่า i เท่ากับ 1
        btn.onPress();  //output ได้ btn1 }
}
```

2.8.5 ข้อแตกต่างระหว่างการประกาศอีเวนต์ทั้ง 2 ตำแหน่ง

หลังจากที่ได้เรียนรู้ไปเรียบร้อยแล้วถึงวิธีการประกาศอีเวนต์ทั้ง 2 ตำแหน่ง ในหัวข้อนี้จะทำความเข้าใจในข้อแตกต่างระหว่าง 2 รูปแบบเพื่อเลือกใช้ตามความเหมาะสม ซึ่งมีดังต่อไปนี้

2.8.5.1 การฝังและคงอยู่ของอีเวนต์ สำหรับตัวอินสแทนซ์

การประกาศอีเวนต์บนตัวอินสแทนซ์นั้นจะเปรียบเสมือนเป็นการฝังโค้ดลงไปในตัวอินสแทนซ์นั้นๆเลย ซึ่งกรณีที่มีการใช้คำสั่งในการสร้างอินสแทนซ์ตัวใหม่ขึ้นมา โดยทำการอ้างอิงตัวอินสแทนซ์ดังกล่าวที่ได้ประกาศอีเวนต์เอาไว้เป็นตัวต้นแบบ จะทำให้ตัวอินสแทนซ์ตัวใหม่นั้นได้รับอีเวนต์ที่ได้ประกาศเอาไว้ไปด้วย และทำงานด้วยอีเวนต์เดียวกันโดยไม่จำเป็นต้องประกาศใหม่

ในขณะที่การประกาศอีเวนต์ให้กับอินสแทนซ์บนไทม์ไลน์นั้นจะให้ผลที่แตกต่างกันคืออินสแทนซ์ที่ถูกสร้างใหม่จะไม่ได้รับอีเวนต์จากตัวต้นแบบไปด้วย โดยจะยึดถือว่าตัวต้นแบบนั้นเป็นอินสแทนซ์ที่ยังไม่มีประกาศอีเวนต์ใดๆเอาไว้ทั้งสิ้น เนื่องจากเป็นการประกาศอีเวนต์ที่เกิดขึ้นภายหลังซึ่งจะมีผลกับอินสแทนซ์ตัวนั้นๆเพียงตัวเดียว ฉะนั้นการที่จะให้อินสแทนซ์

ที่ถูกสร้างใหม่ทำงานด้วยอีเวนต์เดียวกัน จึงจำเป็นต้องทำการประกาศอีเวนต์ด้วยวิธีการเดียวกันกับอินสแทนซ์ที่เป็นตัวต้นแบบ

2.8.5.2 การทำงานที่เกิดขึ้นแบบ หลายอีเวนต์ และ ทำงานเพียงอีเวนต์เดียว

ข้อแตกต่างอีกข้อของการประกาศอีเวนต์บนตัวอินสแทนซ์กับ การประกาศบนไทม์ไลน์จะให้ผลที่แตกต่างกันก็คือ กรณีที่มีการประกาศอีเวนต์ซ้ำๆกันบนอินสแทนซ์ทุกๆอีเวนต์จะทำงานทั้งหมด เช่น ประกาศอีเวนต์ให้บวกค่าตัวแปรใดๆ ทีละ 1 ใน 1 วินาที ถ้าหากประกาศอีเวนต์ซ้ำกัน 2 อีเวนต์ (ชื่ออีเวนต์เดียวกัน) จะทำให้ 1 วินาทีนั้นมีการทำงานอีเวนต์เท่ากับ 2 ครั้งพร้อมๆกัน ซึ่งการทำงานแบบนี้ถือว่าเป็นแบบเกิดขึ้นหลายอีเวนต์ พร้อมๆกัน

แต่สำหรับกรณีที่มีการประกาศอีเวนต์บนไทม์ไลน์ให้กับอินสแทนซ์กรณีที่มีการประกาศอีเวนต์ชื่อซ้ำกันหลายๆอีเวนต์ ผลที่เกิดขึ้นก็คือคำสั่งอีเวนต์ที่ถูกประกาศเอาไว้ก่อนหน้าจะถูกทับด้วยอี เวนต์ใหม่แทน หรือพูดง่ายๆคือ จะทำงานโดยยึดอีเวนต์ใหม่สุดเป็นหลัก ฉะนั้นจะเป็นรูปแบบการทำงานเพียงอีเวนต์เดียวเท่านั้น ซึ่งต่างจากกรณีแรก

2.8.5.3 การเรียกใช้งานแบบฟังก์ชัน

การประกาศอีเวนต์บนตัวอินสแทนซ์ไม่สามารถเรียกใช้งานในรูปแบบฟังก์ชันได้ แต่การประกาศอีเวนต์บนไทม์ไลน์สามารถทำได้ เนื่องจากมีรูปแบบเดียวกับการกำหนดฟังก์ชันให้กับอินสแทนซ์

2.8.6 ข้อแตกต่างของ Button กับ MovieClip เมื่อใช้งานปุ่ม โดยประกาศอีเวนต์

ความแตกต่างระหว่างการประกาศอีเวนต์บนตัวอินสแทนซ์ของ Button และ MovieClip ก็คือ ธรรมชาติของ Button นั้นกรณีที่การระบุพาธ this หรือ _parent ภายในอีเวนต์ฟังก์ชันให้จำไว้เสมอว่า Button นั้นจะมีการอ้างอิงพาธประเภท _parent ให้เอง โดยอัตโนมัติจำนวน 1 ครั้ง ฉะนั้นกรณีที่มี Button ชื่อว่า my_btn จะถูกวางอยู่ภายใต้มูฟวี่คลิปชื่อว่า my_mc

เมื่อทำการประกาศอีเวนต์บนตัวอินสแทนซ์ของ my_btn ดังตัวอย่าง 2.57

ดังตัวอย่าง 2.57

```
on(rollOver) {
    trace(this);  //แสดงผลเป็น my_mc
    trace(_parent); //แสดงผลเป็น _level0
}
```

ซึ่งจะเห็นว่า แทนที่ this จะแสดงผล my_btn เอง แต่กลับแสดง my_mc แทน เนื่องจากว่ามีการอ้างอิงพาธแบบ _parent โดยอัตโนมัติหนึ่งครั้ง (แต่การระบุ _parent.this ไม่สามารถทำได้เองโดยผู้เขียนโปรแกรม) ส่วนการ trace(_parent) ก็จะเหมือนการระบุแบบ this._parent._parent ตามปกติในมูฟวี่คลิป ฉะนั้นจะได้ผลลัพธ์เท่ากับ _level0 ซึ่งเป็นระดับสูงสุดซึ่งก็คือไทม์ไลน์หลักของ swf ชิ้นนั้นแล้ว

สำหรับกรณีที่ประกาศโค้ดบนตัว MovieClip จะได้ผลลัพธ์ดังตัวอย่าง 2.58

ตัวอย่าง 2.58

```
on(rollOver) {
    trace(this);  //แสดงผลเป็น my_btn
    trace(_parent); //แสดงผลเป็น my_mc
            }
```

2.8.7 ปัญหาอีเวนต์บังอีเวนต์

สำหรับข้อนี้เป็นปัญหาที่สำคัญที่หลายๆคนมักจะพบเจอแต่ไม่ทราบว่าเกิดอะไรขึ้น จึงทำให้อินสแทนซ์ดังกล่าวไม่สามารถทำงานได้ถึงแม้ว่าจะประกาศอีเวนต์ไว้เรียบร้อยแล้วก็ตาม ตัวอย่างเช่น มี mc2 อยู่ภายใน mc1 ดังนี้

กรณีที่ประกาศอีเวนต์ให้กับ mc1 และ mc2 ทั้งคู่ ดังตัวอย่าง 2.59

ตัวอย่าง 2.59

```
mc1.onPress = function() {
    trace(this); //output _level0.mc1;
    this._alpha = 50;  //ปรับค่าความโปร่งใส่เท่ากับ 50% }
```

โดยจากโค้ดดังกล่าวนั้น เมื่อทำการคลิกตัว mc2 แล้ว หน้าต่าง output ควรจะแสดงพาธตัวมันเอง แต่ในความเป็นจริงนั้นคือ เมื่อทำการกด mc2 หน้าต่าง output จะไม่แสดงผลว่าเกิดการคลิก mc2 แต่จะมองว่า mc2 เป็นเพียงแค่ส่วนหนึ่งของ mc1 เท่านั้น ฉะนั้นเมื่อคลิก mc2 ก็จะแสดงอีเวนต์ของ mc1 แทน ซึ่งจะได้ผลลัพธ์ก็คือ _level0.mc1 แทน

2.8.7.1 วิธีแก้ปัญหาอีเวนต์บังอีเวนต์

วิธีแก้สำหรับปัญหานี้ก็คือ การนำอีเวนต์ไปประกาศที่ระดับชั้นเดียวกัน โดยทำการสร้างพื้นที่สำหรับมูฟวี่คลิประดับชั้นนอกโดยนำมาวางไว้ที่ระดับ เดียวกับมูฟวี่คลิปในระดับชั้น เพื่อไม่ให้เกิดการบังเกิดขึ้น ส่วนการอ้างอิงพาธก็ให้เปลี่ยนไปใช้การอ้างอิงแบบ absolute (_root) หรือ relative (this,_parent) เข้ามาช่วยในการอ้างอิงแทน

สร้างมูฟวี่คลิปชื่อว่า area_mc ขึ้นมาเป็นพื้นที่ของ mc1 โดยเฉพาะ หลังจากจึงทำการกำหนดอีเวนต์ให้กับ mc1 ใหม่ดังตัวอย่าง 2.60

ตัวอย่าง 2.60

```
mc1.area_mc.onPress = function() { trace(this);  //output _level0.mc1.area_mc;
                                   trace(this._parent); //output _level0.mc1;
                                   this._parent._alpha = 50; //ปรับค่าความโปร่งใส่ 50% }
mc1.mc2.onPress = function() {  trace(this); //output _level0.mc1.mc2;
                                  this._alpha = 50;  //ปรับค่าความโปร่งใส่เท่ากับ 50% }
```

จะเห็นว่าผลลัพธ์เหมือนกันเมื่อคลิกที่ mc1 แต่จะต่างกันที่ mc2 ก็สามารถคลิกกดได้ด้วย เนื่องจากไม่เกิดปัญหาอีเวนต์บังอีเวนต์เพราะได้กำหนดอีเวนต์ให้กับวัตถุ ที่อยู่ระดับชั้นเดียวกัน

2.8.8 การกำหนดอีเวนต์ด้วย Listeners

ในเวอร์ชันหลังๆนั้น จะมีรูปแบบการกำหนดอีเวนต์ด้วย listener เพิ่มขึ้นมา และจะมีการใช้รูปแบบนี้เป็นวิธีการหลักใน AS3 อีกด้วยเพียงแต่จะต่างออกไปตรงชื่อคำสั่งนิดหน่อย

สำหรับการทำความเข้าใจในเรื่องของ listener นั้นสามารถทำความเข้าใจได้ไม่ยาก โดยให้นึกถึง listener เปรียบเสมือนตัวแทนสำหรับการรับฟังขึ้นมา ซึ่งมีหน้าที่เอาไว้ฟังอีเวนต์ต่างๆ และวัตถุใดๆที่ได้รับ listener นั้นไว้ ก็จะทำงานตามที่ listener ได้รับคำสั่งหรืออีเวนต์ที่เกิดขึ้น สำหรับรูปแบบการประกาศ อีเวนต์ listener จะเป็นดังตัวอย่าง 2.61

ตัวอย่าง 2.61

```
my_listener = new Object();  //สร้าง listener ขึ้นมาเองด้วยชื่ออะไรก็ได้
my_listener.onEventListener = function()
{ //ประกาศอีเวนต์ให้กับ listener โดยชื่ออีเวนต์ต้องเป็นอีเวนต์ที่ Object หรือ Class รองรับ }
Object.addListener(my_listener);  //ประกาศอีเวนต์ listener ให้กับ Object หรือ Class ที่ต้องการ
```

สำหรับ Class ที่สามารถกำหนด Listeners ได้จะมี Mouse Keyboard Stage MovieClipLoader (สำหรับการโหลดไฟล์), Selection นอกจากนี้จะยังสามารถพบการประกาศรูปแบบนี้ได้ใน component อีกด้วย เพียงแต่จะใช้คำสั่ง addEventListener แทน (ใช้คำสั่งนี้ใน AS3 เช่นกัน) โดยการกำหนดอีเวนต์รูปแบบนี้ถือว่าเป็นการประกาศอีเวนต์โดยปกติอยู่แล้ว ฉะนั้นจะถือว่าเป็นอีเวนต์เฉยๆไม่ได้เรียกว่า Listeners

เพื่อเพิ่มความเข้าใจในการใช้งานที่มากขึ้น ให้ศึกษาจากตัวอย่างดังต่อไปนี้ ซึ่งเป็นตัวอย่างการประกาศอีเวนต์ listener ให้กับ Mouse

ให้ประกาศโค้ดดังกล่าวที่ไทม์ไลน์ ดังตัวอย่าง 2.62

ดังตัวอย่าง 2.62

```
my_ls = {};
//สร้าง listener ขึ้นมาจาก Object โดยเขียนในรูปแบบย่อคือ {} ซึ่งแทน Object
my_ls.onMouseDown = function() { //ทำเมื่อคลิกเมาส์
        trace("onMouseDown"); }
my_ls.onMouseMove = function() { //ทำเมื่อปล่อยเมาส์
        trace("onMouseMove"); }
my_ls.onMouseUp = function() { //ทำเมื่อปล่อยเมาส์
        trace("onMouseUp"); }
my_ls.onMouseWheel = function() { //ทำเมื่อหมุน mouse wheel หรือล้อหมุน
        trace("onMouseWheel"); }
Mouse.addListener(my_ls); //ใส่ listener ให้กับ Mouse
```

ภายในเรื่องนี้ ได้ทำการศึกษาอีเวนต์ในรูปแบบต่างๆทั้งหมดเรียบร้อยแล้ว ซึ่งการศึกษาและจดจำอีเวนต์ของคลาสต่างๆเป็นสิ่งที่สำคัญมาก เพราะจะช่วยให้การทำงานของยืดหยุ่นมากขึ้นและเลือกใช้คลาสได้อย่างเหมาะสม

บทที่ 3
โมเดล 3 มิติ และการนำมาใช้

3.1 การสร้างโมเดล 3 มิติ

3.1.1 ออโต้เดสค์ทรีดีเอสแมกซ์(Autodesk 3ds Max)

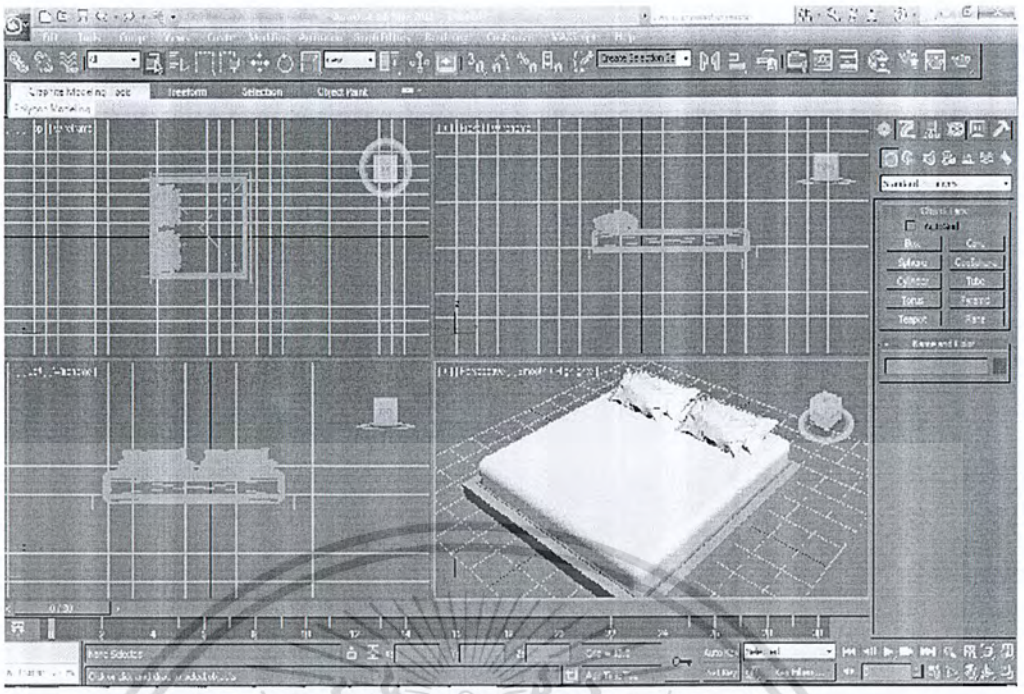
โปรแกรมทรีดีเอสแมกซ์(3ds Max)เป็นหนึ่งในโปรแกรมที่รู้จักกันอย่างกว้างขวางในเชิงพาณิชย์ เป็นโปรแกรมที่สามารถสร้างภาพได้สมจริง ทั้งในเรื่องรายละเอียดของขอบเส้น การจัดแสงเงา การปรับความเข้มแสง และมุมกล้อง และสามารถเรนเดอร์ภาพออกมาได้สวย สมจริง โปรแกรมสามารถประยุกต์ใช้ได้หลายประเภท มีเครื่องมือที่หลากหลาย ใช้งานง่าย และสามารถหาข้อมูล ศึกษาวิธีการใช้โปรแกรมได้มากมาย มีตัวช่วยให้งานออกมาสวยโดยปลั๊กอินซึ่งก็มีให้เลือกใช้มากมาย

โปรแกรมทรีดีเอสแมกซ์ได้พัฒนาโดยบริษัทออโต้เดสค์อิงค์(AutoDesk Inc.) โดยลักษณะพิเศษของโปรแกรมทรีดีเอสแมกซ์ช่วยให้สามารถสร้างโมเดลได้ง่ายและใส่พื้นผิวให้โมเดลอย่างสมจริง ในส่วนของภาพเคลื่นไหว 3 มิติ(3D Animation) จะสามารถควบคุมการเคลื่อนที่ของตัวละครเหมือนดังมีชีวิต จึงนิยมทำงาน 3 มิติ ไปใช้ในงานโฆษณาและงานภาพยนตร์

ลักษณะพิเศษในการสร้างโมเดล 3 มิติ สามารถควบคุมมุมมองของวัตถุได้อย่างอิสระ โดยการหมุนโมเดลให้เห็นตำแหน่งของ วัตถุทุก ๆ ด้านสำหรับปรับแต่งโครงสร้างโมเดลและเมื่อสร้างโมเดลแล้วให้ใส่พื้นผิดและ จัดแสงให้แก่โมเดลให้เป็นธรรมชาติ

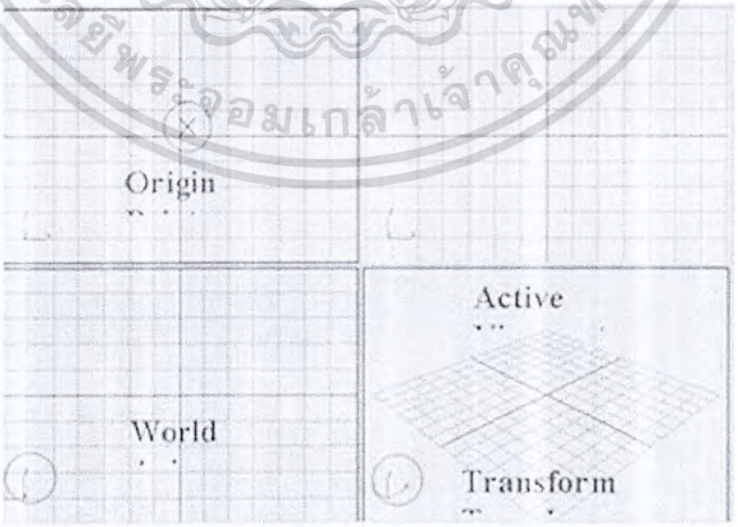
3.1.1.1 ส่วนประกอบของโปรแกรมทรีดีเอสแมกซ์

โปรแกรมทรีดีเอสแมกซ์เป็นโปรแกรมที่มีความโดดเด่นในการสร้างภาพ 3 มิติ และงานสร้างอนิเมชัน(Animation) เหตุที่โปรแกรมได้รับความนิยม เนื่องจากสามารถขึ้นรูปโมเดลได้ง่ายและความสามารถของโปรแกรมก็ครอบคลุมการทำ งานได้เกือบทุกเรื่องของงานอนิเมชัน ส่วนประกอบและเครื่องมือที่ใช้ในการทำงานของโปรแกรมทรีดีเอสแมกซ์เพื่อเป็นพื้นฐานในการงานงาน 3 มิติ ต่อไป ดังนี้



รูป 3.1 ลักษณะของโปรแกรมทรีดีเอสแมกซ์

ในการทำงานด้วยโปรแกรมทรีดีเอสแมกซ์จะเริ่มจากการขึ้นรูปโมเดล ด้วยกลุ่มเครื่องมือ Rollout สร้างรูปขึ้นมา จากนั้นปรับรูปทรงด้วยชุดคำสั่ง Object Categories ต่อด้วยสร้างอนิเมชัน จากกลุ่มเครื่องมือ Reactor เป็นเครื่องมือที่ช่วยในการทำภาพเคลื่อนไหวให้กับโมเดลและ Time Slider กำหนดช่วงเวลาการแสดงผลงาน ส่วนกรณีที่ต้องการเรียกใช้เครื่องมือกลุ่มใด ให้คลิกขวาบริเวณ Main Toolbar



รูป 3.2 ส่วนประกอบของวิวพอร์ต

3.1.1.2 พื้นฐานการใช้งานโปรแกรมทรีดีเอสแมกซ์

สำหรับผู้ที่ต้องการศึกษาโปรแกรมทรีดีเอสแมกซ์ต้องเริ่มจากการศึกษาตั้งแต่เครื่องมือพื้นฐานเกี่ยวกับการควบคุมวัตถุ เช่น ย่อ/ขยาย หมุนวัตถุและส่วนที่สำคัญอีกประการหนึ่งคือ การปรับมุมมองของภาพ การใช้เส้นกริด ตำแหน่งการวางวัตถุ Pivot Points ซึ่งพื้นฐานเหล่านี้ถือว่าเป็นจุดเริ่มต้นที่สำคัญในการสร้างงานอนิเมชันเป็นอย่างมาก

3.1.1.2.1 ปรับมุมมองของโมเดลบนวิวพอร์ต

การสร้างโมเดลในวิวพอร์ตสามารถทำงานควบคู่กับวิวพอร์ตคอลโทรล (Viewport Controls) ซึ่งได้รับรวบรวมคำสั่งในการย่อ/ขยายหรือหมุนในหน้าจอวิวพอร์ตและสามารรถปรับแต่งโมเดลได้ง่าย โดยตำแหน่งวิวพอร์ตคอลโทรลจะอยู่ด้านล่างมุมซ้ายของโปรแกรม ซึ่งมีรายละเอียดดังนี้

Zoom ใช้ย่อ - ขยายหน้าจอวิวพอร์ตเพื่อใช้ดูรายละเอียดของตัวโมเดล การใช้งานเพียงคลิกเมาส์แล้วลากขึ้นเป็นการขยายภาพ หรือลากเมาส์ลงเป็นการย่อขยายภาพ (อาจใช้ลูกกลิ้งของเมาส์เลื่อนเพื่อย่อ/ขยายหน้าจอในวิวพอร์ตได้)

Zoom All เป็นการย่อขยายวิวพอร์ตทั้งหมดทุกด้านคือ Top Front Left และ Prespective โดยย่อขยายพร้อมกันทุก ๆ ด้าน

Zoom Extents เป็นการย่อหรือขยายโมเดลให้เต็มหน้าวิวพอร์ต

Field of View ขยายภาพด้วยวิธีการแดรกเมาส์ในวิวพอร์ต

Pan View เป็นการเลื่อนภาพในวิวพอร์ตเพื่อดูรายละเอียดโมเดล(เป็นคำสั่งที่สำคัญต่อการทำงานเป็นอย่างมาก)

Arc Rotate เป็นการหมุนวิวพอร์ตเพื่อหาตำแหน่งที่จะใช้แก้ไขวัตถุ โดยเมื่อเลือกคำสั่ง Arc Rotate นั้น โปรแกรมจะสร้างจุดศูนย์กลางในการหมุนวัตถุ

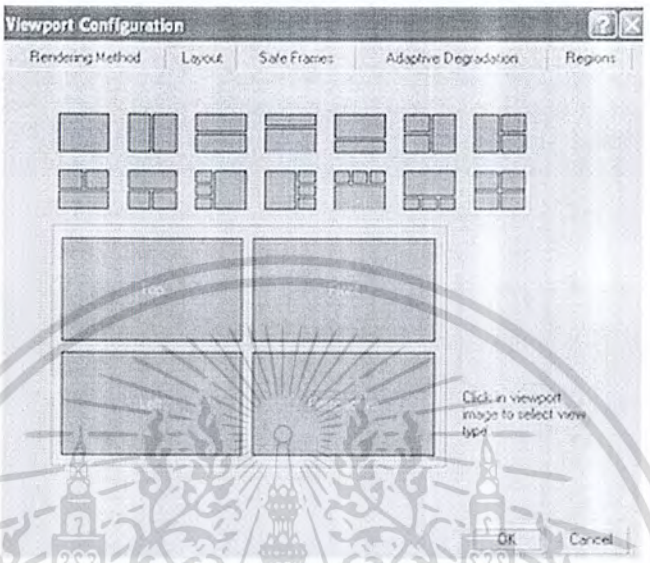
Min/Max Toggle ใช้ปรับหน้าจอของวิวพอร์ตให้แสดงผลเต็มหน้าจอหรือให้เลือกเฉพาะบางวิวพอร์ตก็สามารถทำได้ ซึ่งจะเหมาะสำหรับการปรับแต่งหรือแก้ไขรูปทรง (กรณีที่ต้องการให้วิวพอร์ตกลับสู่สถานะปกติให้คลิกซ้ำที่คำสั่ง Min/Max Toggle อีกครั้ง

การปรับมุมมองในวิวพอร์ตสามารถใช้คีย์ลัดช่วยในการทำงานได้ด้วย เพราะจะช่วยให้ทำงานได้สะดวกขึ้น สำหรับเครื่องมือที่จำเป็นต่อการทำงาน ได้แก่ ขยายวัตถุด้วยคำสั่ง Zoom หมุนวัตถุด้วยคำสั่ง Arc Rotate และย่อ/ขยายวิวพอร์ตให้แสดงเฉพาะส่วนด้วยคำสั่ง Min/Max Toggle

3.1.1.2.2 การปรับแต่งวิวพอร์ต

หน้าจอของวิวพอร์ตสามารถที่จะปรับแต่งได้ด้วยตนเองว่า ในหน้าจอส่วนใดที่ต้องการให้แสดงรายละเอียดของโมเดลในขนาดใหญ่ หรือเล็กตามความต้องการของผู้ใช้งาน

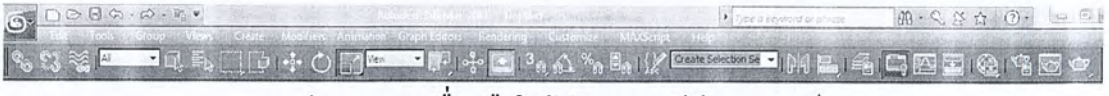
วิวพอร์ตนอกจากกำหนดค่าด้วยตัวเองแล้วยังสามารถกำหนดวิวพอร์ตตามแบบฟอร์มของโปรแกรมที่มีให้เลือกใช้งานอยู่หลายรูปแบบ ซึ่งวิธีการเรียกใช้วิวพอร์ตคอนฟิกกูเรชัน สามารถเรียกใช้งานได้โดยคลิกขวาบริเวณวิวพอร์ตคอลโทรล



รูป 3.3 หน้าต่างของวิวพอร์ตคอนฟิกกูเรชัน

3.1.1.2.3 แถบเครื่องมือของโปรแกรมทรีดีเอสแมกซ์

ทำความรู้จักเครื่องมือที่ สำคัญในโปรแกรมทรีดีเอสแมกซ์ซึ่งแถบเครื่องมือหลักของโปรแกรม คือ Main Toolbar และ Reactor Toolbar เป็นคำสั่งพื้นฐาน ในส่วนของ Main Toolbar นั้นจะมีเครื่องมืออยู่จำนวนมาก ฉะนั้นการเรียกใช้งานจึงต้องใช้ Scrolling เลื่อนดูคำสั่งไปด้านซ้ายมือ โดยจะมาเรียนรู้คำสั่งต่าง ๆ ของโปรแกรมเพื่อนำไปประยุกต์ใช้ในการสร้างโมเดล



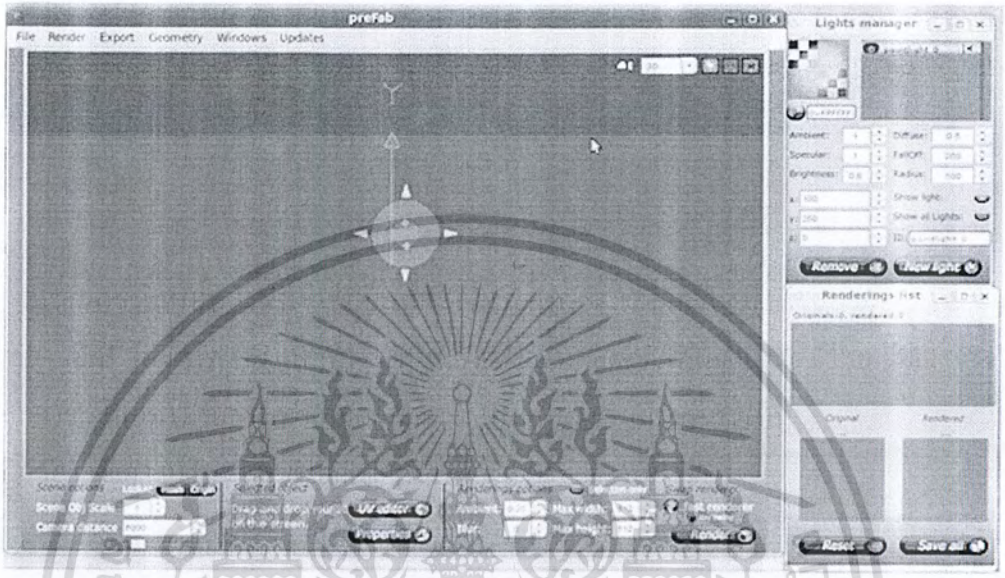
รูป 3.4 แถบเครื่องมือในโปรแกรมทรีดีเอสแมกซ์

3.2 วิธีการนำเข้าโมเดลนามสกุล .3DS ด้วยพรีเฟบ(preFab) และอเวย์ทรีดี(Away3D)

เมื่อใช้โปรแกรมทรีดีเอสแมกซ์ในการสร้างโมเดล 3 มิติ ขึ้นมาแล้ว ให้ทำการแปลงไฟล์ไปเป็น .3ds เพื่อที่จะนำมาใช้ เมื่อได้ไฟล์ โมเดล 3มิติ ที่เป็น .3ds มาแล้ว ขั้นต่อไปจะทำการใส่ลวดลายให้กับโมเดลสามมิตินั้น ซึ่งการใส่ลวดลายให้กับวัตถุด้วยอเวย์ทรีดีนั้นเป็น Flash Platform ที่เขียน

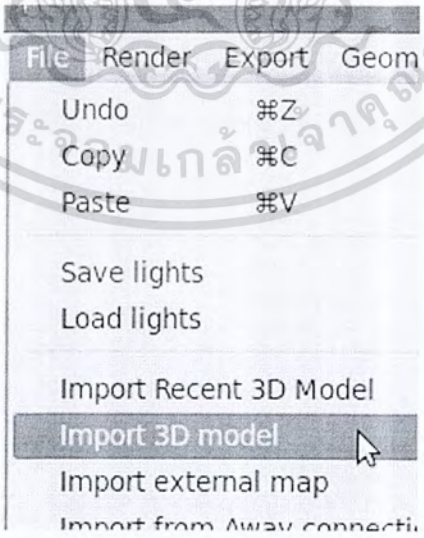
ด้วยภาษาแอกชันสคิปท์ดังนั้นจึงจำเป็นต้องใช้พรีเฟบ ซึ่งเป็นเครื่องมือที่จะช่วยสร้างคลาสของโมเดลสามมิติพร้อมลวดลาย แทนที่จะมานั่นเล็งเอาเองพล็อตจุดแทนจุดเอาเอง โดยมีวิธีการดังนี้

1) ดาวน์โหลดโปรแกรมพรีเฟบก่อน ซึ่งพรีเฟบเป็นโปรแกรมที่ทำงานบน Adobe AIR ถ้าต้องการดาวน์โหลดพรีเฟบมาติดตั้งจะต้องลง Adobe Air ก่อน



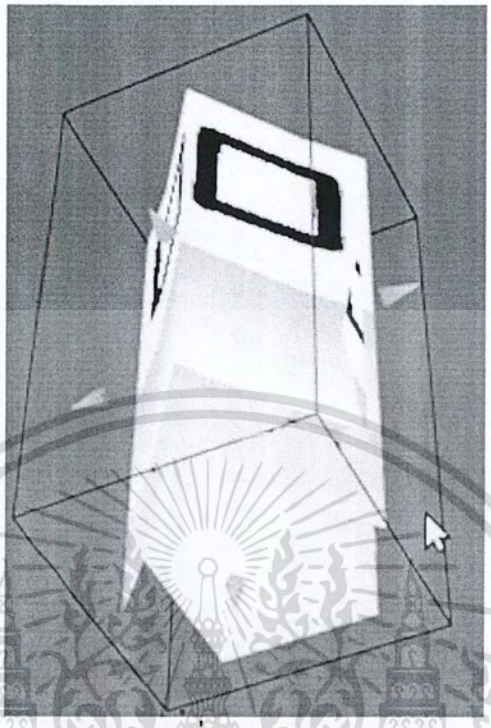
รูป 3.5 หน้าต่างโปรแกรมพรีเฟบ

2) ไปที่ File > Import 3D model ดังรูป



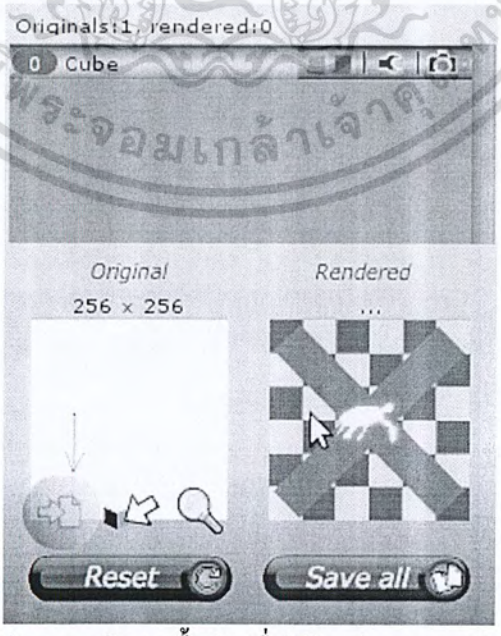
รูป 3.6 อธิบายวิธีการอิมพอร์ตไฟล์

3) โมเดลที่ทำใน Blender ก็จะมาปรากฏบนพรีเฟบ เหมือนในรูป



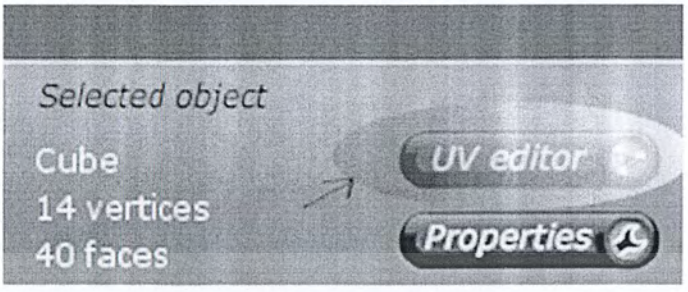
รูป 3.7 โมเดลที่โหลดเข้ามาในพรีเฟบ

4) เมื่อโหลดโมเดลมาแล้วต่อไปจะโหลดพื้นผิวที่จะมาไว้บนวัตถุ สังเกตุที่ Renderings list จะมีที่ให้โหลด ดูในรูปด้านล่าง จากนั้นก็เลือกภาพที่ต้องการ



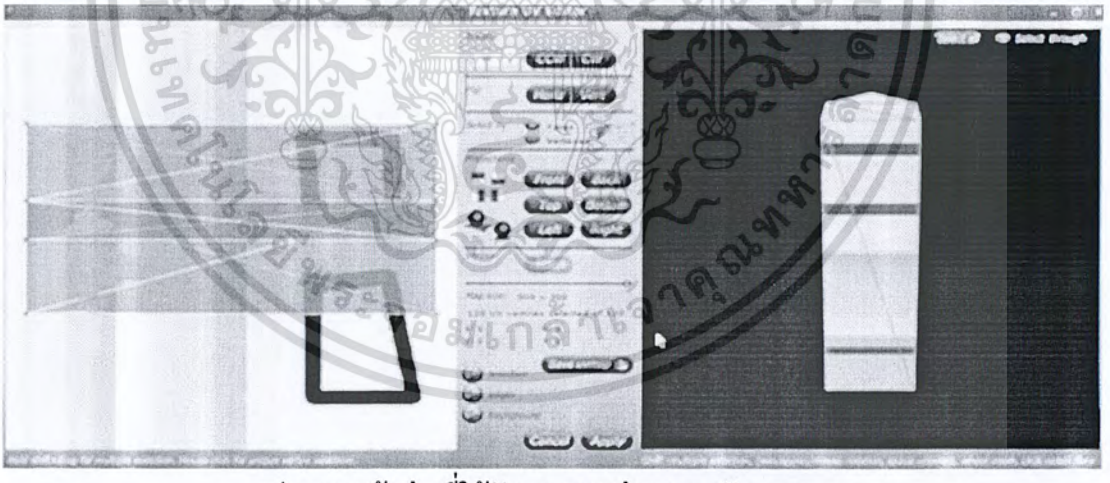
รูป 3.8 การโหลดพื้นผิวเพื่อให้มาปรากฏบนวัตถุ

5) เมื่อโหลดภาพมา ตอนนี้โมเดลก็จะมีพื้นผิวแล้ว แต่เป็นลายแปลกๆ เนื่องจากตำแหน่งที่วางอยู่อาจไม่ได้เป็นไปตามที่ต้องการ จึงต้องทำการจัดการกับพื้นผิวนั้น โดยเลือกที่ UV editor



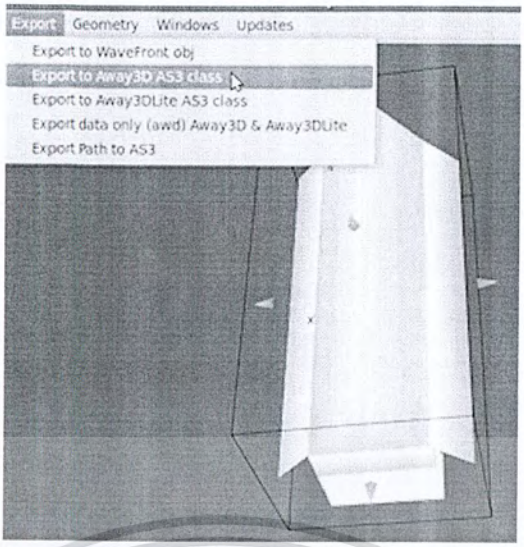
รูป 3.9 หน้าจอแสดง UV editer

6) จะปรากฏหน้าต่างมาให้ เพื่อทำการจัดการระหว่าง face กับ picture ที่โหลดมา โดยหน้าจอทางขวา เป็นส่วนของการ preview และเลือกวัตถุ กด Ctrl ค้าง แล้วลากเพื่อหมุน ส่วนหน้าจอทางซ้าย เป็นส่วนของ face ที่ถูกทำการเลือกโดยหน้าจอทางขวา เพื่อทำการแก้ไข หน้าจอพาเนลตรงกลาง เป็นส่วนของการ rotate,flip ฯลฯ กับ vertex ของ face ที่ถูกเลือกจากหน้าจอทางซ้ายอีกที



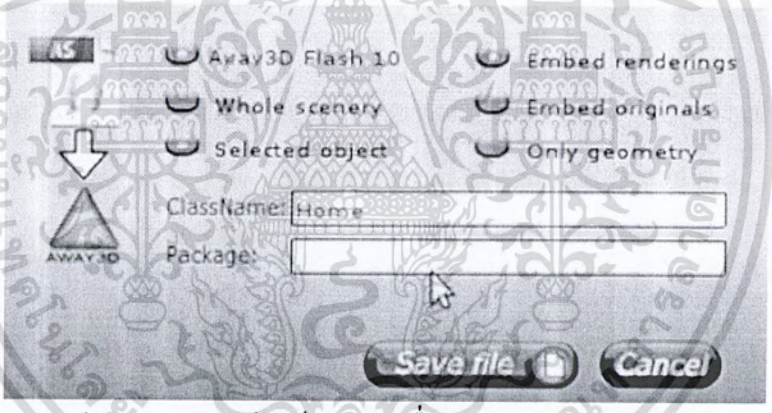
รูป 3.10 หน้าต่างที่ใช้จัดการระหว่าง face กับ picture

7) จากนั้นเมื่อจัดภาพกับโมเดลจนเป็นที่พอใจแล้วก็ ไปที่ Export>Export to Away3D AS3 class เพื่อบันทึกภาพที่ทำอยู่ ออกไปเป็นไฟล์ .as



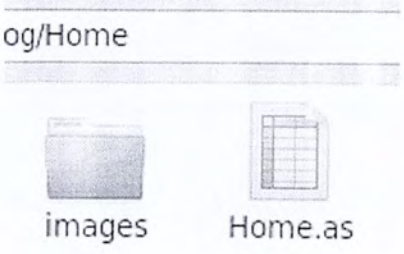
รูป 3.11 ทำการเอ็กซ์พอร์ตไฟล์ ไปเป็นนามสกุล .as

8) ใส่ชื่อและ package ของคลาส



รูป 3.12 การ save ไฟล์ โดยระบุชื่อและ package ของคลาส

9) เมื่อเสร็จขั้นตอนที่ 8 ตอนนี้จะต้องได้ไฟล์.as แล้วก็โฟล์เดอร์



รูป 3.13 เมื่อทำการเอ็กซ์พอร์ตตามขั้นตอนจะได้ไฟล์ .as และ โฟล์เดอร์

3.3 การนำข้อมูลจากภายนอกมาใช้อย่างมีประสิทธิภาพโดยคำนึงถึงขนาดและความเร็ว

มีไฟล์ 3 มิติ ไม่กี่ประเภทที่สามารถเพิ่มประสิทธิภาพให้สามารถใช้งานได้บนเว็บไซด์หรือกับแฟลชไฟล์บางประเภทนั้นไม่ได้ถูกออกแบบให้การใช้แบนด์วิดท์เป็นไปได้ตามต้องการ เช่น ไฟล์ COLLADA ส่วนไฟล์อื่นๆ อาจจะต้องใช้การทำไบนารี เช่นการทำ ZIP ที่ต้องมีการทำการประมวลผลพิเศษเพื่อขยายขนาด ชะลอตัวลงในระยะเวลาของการวิเคราะห์ ด้วยเหตุผลที่กล่าวมานี้ อเวย์ทรีดีจึงเสนอให้มีการแปลงไฟล์ที่สามารถแปลงโมเดล 3 มิติ ให้สามารถใช้งานได้ในอเวย์ทรีดี และส่งข้อมูลออกมาในรูปแบบของคลาสภาษาแอกชันสคิปท์กระบวนการในคลาส AS3Exporter อยู่ใน away3d.exporters แพ็คเกจ และใช้กระบวนการไม่กี่ขั้นตอนในการแปลงไฟล์

1) โหลดไฟล์ 3ds COLLADA หรือโมเดลแบบอื่นๆ ในอเวย์ทรีดี ลงในคลาส Loader3D
2) ใช้คลาส AS3Exporter ในการแปลงโมเดลโดยใช้ภาษาแอกชันสคิปท์และส่งไปยังคลิปบอร์ด
3) สร้างคลาสใหม่ในโฟลเดอร์โปรเจค และใส่เนื้อหาของคลิปบอร์ดลงในคลาส
4) คอมไพล์โปรเจคอีกครั้ง จะเป็นการสร้างโมเดล โดยการยกตัวอย่างสร้างคลาสใหม่ แทนการโหลดไฟล์ที่แยกออกมา

เมื่อได้ดำเนินการตามขั้นตอนจะทำให้สามารถนำโมเดลมาใช้ได้ในไฟล์แอปปลิเคชัน SWF หรือจะคอมไพร์ SWF แยกออกมาก็ได้ ที่สามารถเรียกใช้ได้อย่างปกติในคลาส Flash Loader ไฟล์ SWF จะถูกบีบอัดโดยใช้ ZIP และมีข้อดีคือทำให้ไฟล์มีขนาดเล็ก รวมกับการเรียกคืนข้อมูลที่ถูกบีบอัดในรูปแบบปกติสำหรับการใช้งานที่เกิดขึ้นใน Flash Player ดังนั้นจึงทำให้มันทำงานได้เร็วมาก เช่นเดียวกับในการแยกวิเคราะห์ข้อมูลของโมเดลที่มีประสิทธิภาพสูง เพราะว่าวิธีการเพิ่มประสิทธิภาพที่มีอยู่ในการสร้างคลาสดังกล่าว

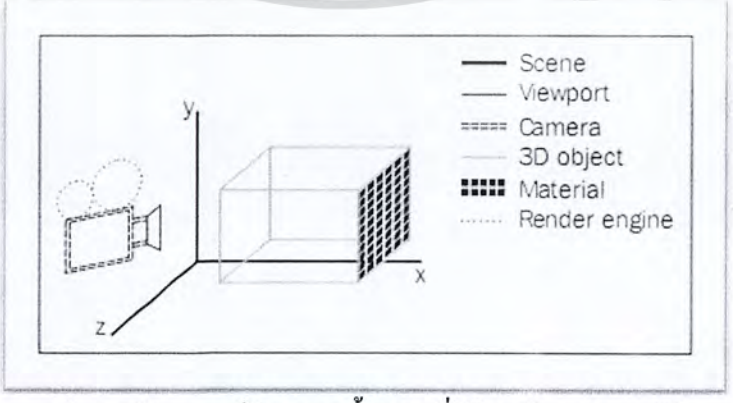
บทที่ 4
พื้นฐาน Away 3D

การโปรแกรมแฟลชเป็นเพจฟอร์มที่มีอิสระมากกว่าเพจฟอร์มอื่นๆ ซึ่งมีการใช้งานมายาวนาน และหลังจากที่เปิดตัวเวอร์ชั่น CS4 Adobe ได้เพิ่มความสามารถในการทำ 3 มิติเข้ามาด้วย แต่การทำ 3 มิติ ไม่ได้เริ่มมาพร้อมกับภาษาแอกชันสคิปท์เวอร์ชัน 3 ดังนั้นนักพัฒนาแฟลชจึงร่วมกันพัฒนามาช้านาน อย่างไลบรารีของอเว์ยทรีดีเกิดจากความพยายามของนักพัฒนาที่เขียนขึ้นมาด้วยโครงสร้างเชิงวัตถุ ซึ่งได้รับการยอมรับและมีความนิยมเป็นอย่างมากทำให้อเว์ยทรีดี เติบโตมาจนถึงทุกวันนี้และมีกลุ่มนักพัฒนาที่มีศักยภาพสูงจากทั่วทุกมุมโลกให้ความสนใจ และมีการพัฒนาอย่างต่อเนื่องมาจนถึงทุกวันนี้

ก่อนที่จะเริ่มต้นเขียนอเว์ยทรีดีในรูปแบบของแฟลชจะตั้งมีการตั้งค่าเริ่มต้นที่ถูกต้อง ขั้นตอนการติดตั้งที่ถูกต้องจะขึ้นอยู่กับ workflow แบ่งเป็นสองขั้นตอนเสมอ คือ ขั้นตอนแรกได้รับรหัสมาจากนั้นให้ทำการตั้งค่าพัฒนาสภาพแวดล้อมในตัว (IDE) ซึ่งใช้กับอเว์ยทรีดีสำหรับ source code ให้หาโปรแกรม โดยให้มั่นใจว่าเป็นรุ่นใหม่ล่าสุด ในที่นี้จะกล่าวถึงการตั้งค่าต่างๆ ก่อนเริ่มการเขียนโปรมแกรม จึงจะเป็นการดีที่สุดที่จะปฏิบัติตามการตั้งค่า ซึ่งเกี่ยวข้องกับการแก้ไข

การใช้งานไลบรารีในอเว์ยทรีดี เป็นไลบรารี source code ซึ่งมีอยู่ประมาณ 400 ไลบรารี หรือเช่นเดียวกับในภาษาแอกชันสคิปท์เวอร์ชัน 3.0 ซึ่งใช้งานได้เหมือนกับไฟล์ที่ได้จากการเขียนขึ้นมาเองและไฟล์เหล่านี้ต้องถูกรวมเข้าไปกับโปรเจคที่สร้างขึ้นก่อน ถึงจะสามารถทำงานได้

การเขียนโปรแกรมสำหรับภาพ 3 มิติ ความรู้เบื้องต้นที่จำเป็นจะต้องศึกษาเกี่ยวกับระบบสามมิติในอเว์ยทรีดีเพื่อให้เข้าใจในการเขียนโปรแกรม การที่จะเขียนโปรแกรมสำหรับวัตถุทุกประเภทจะอธิบายเป็นองค์ประกอบที่ใช้งานอยู่ในโมเดลสามมิติตามลักษณะที่มองเห็นวัตถุทั้งหมดดังนี้



รูป 4.1 อธิบายองค์ประกอบทั้งหมดที่ใช้งานในงานสามมิติ

4.1 สร้าง 3 มิติ โปรเจค

4.1.1 การเขียนโปรแกรมเบื้องต้น

ขั้นตอนแรกที่เกี่ยวกับอเว์ย์ทรีดี ซึ่งเป็นเครื่องมือที่มีประสิทธิภาพแบบเรียลไทม์ สำหรับ Flash หรือ เฟล็กส์(Flex) ซึ่งเป็นการอธิบายวิธีพื้นฐานของคลาสในอเว์ย์ทรีดีแบบบรรทัดต่อบรรทัด เพื่อให้สามารถยังคงใช้ภาษาแอกชันสคิปท์การมีพื้นฐานของงานออกแบบจะสามารถเข้าใจสิ่งที่เหลือของเนื้อหานี้ ไม่ว่าสิ่งใดที่ต้องการจะเขียนในอเว์ย์ทรีดีก็มีบางอย่างที่มักจะต้องตั้งค่าก่อนค่าพื้นฐานจะอธิบายเกี่ยวกับพื้นฐาน ฉาก มุมมอง กล้อง พื้นผิว และศึกษาบางส่วนของความเป็นไปได้ ตัวอย่างเช่นในแต่ละบท จะได้รับการจัดเป็นคลาสภาษาแอกชันสคิปท์ จึงสามารถนำมาใช้ในทั้ง Flash และ เฟล็กส์

4.1.1.1 พื้นฐานของคลาสในอเว์ย์ทรีดี

เป็นพื้นฐานของการเริ่มต้นการใช้กล้องและขั้นตอนต่างๆในอเว์ย์ทรีดีถ้าพอจะมีพื้นฐานเกี่ยวกับการเขียนภาษาแอกชันสคิปท์อาจจะข้ามไปอ่านในเรื่องต่อไป แต่ตอนนี้จะอธิบายโค้ดแบบเป็นบรรทัดต่อบรรทัด ดังตัวอย่าง 4.1

ตัวอย่าง 4.1

```
package {
    import away3d.containers.View3D;
    import away3d.primitives.Sphere;
    import flash.display.Sprite;
    public class Basic01 extends Sprite {
        public function Basic01() {
            var view:View3D = new View3D({x:200,y:200});
            addChild(view);
            var sphere:Sphere = new Sphere();
            view.scene.addChild(sphere);
            view.render();
        }
    }
}
```

บรรทัดที่ 1 ในภาษาแอกชันสคิปท์เวอร์ชัน 3 จำเป็นที่จะต้องให้คลาสทั้งหมดนั้นอยู่ในแพคเกจ เพื่อความง่ายใช้แพคเกจที่ไม่มีชื่อ (ที่เรียกว่าแพกเกจเริ่มต้น)

บรรทัดที่ 2-4 เพื่อให้สามารถใช้อเวย์ทรีดีจำเป็นต้องนำคลาสที่มีฟังก์ชั่นที่ต้องการใช้เข้ามา เช่นเดียวกับคลาสมาตรฐาน Sprite

บรรทัดที่ 5 ชื่อของคลาสตัวอย่าง เป็นคลาส extends ซึ่งจะเป็นคลาสที่มีคุณสมบัติ และสามารถทำงานได้เหมือนกับคลาส Sprite

บรรทัดที่ 6 ฟังก์ชั่นที่มีชื่อเหมือนกับคลาสจะเรียกว่าฟังก์ชั่นคอนสตรัคเตอร์ทุกครั้งที่มีการเรียกใช้คลาสเกิดขึ้น ภายในฟังก์ชั่นนั้นๆ ก็จะถูกดำเนินการ

บรรทัดที่ 7-8 เป็นการสร้างมุมมองในสามมิติ เป็นการสร้างภาพในมุมมองเสมือนจริง โดยมีตัวแปร x และตัวแปน y เป็นตัวบอกจุดศูนย์กลางในโลก 3 มิติ ของอเวย์ทรีดี จากนั้นจะเพิ่มมุมมองในคลาส จึงจะปรากฏภาพในระยะแฟลช

บรรทัดที่ 9-10 สร้างทรงกลม โดยเพิ่มทรงกลมเพื่อเริ่มต้นภาพ 3 มิติ ในมุมมองของภาพ (เช่นเดียวกับการแสดงวัตถุอื่นๆ ในแฟลชมีเพียงสิ่งที่อยู่ในฉากเท่านั้นที่สามารถมองเห็นได้) เนื่องจากไม่ผ่านในพารามิเตอร์ในตำแหน่งใด ๆ สำหรับ X Y และ Z ทรงกลมจะถูกสร้างขึ้นที่จุดศูนย์กลางของมุมมอง (X = 0, Y = 0, Z = 0)

บรรทัดที่ 11 ในอเวย์ทรีดีสามารถออกแบบได้ว่าจะแสดงผลมุมมองแบบใด การแสดงผลเป็นกระบวนการของการแปลงภาพสามมิติเสมือนจริง เป็นภาพที่สามารถดูได้ด้วยสองมิติ การแสดงมาจากกระบวนการของ CPU และความสามารถในการตัดสินใจ เมื่อมีการเปลี่ยนแปลงค่าจะมีประโยชน์อย่างมาก การทำงานที่ดีคือการแสดงผลมุมมอง เมื่อผู้ใช้มีการเลื่อนเมาส์ในแฟลช

ข้างต้นนี้เป็นตัวอย่างพื้นฐานอย่างคร่าวๆ ที่นำมาแสดงให้ดูเป็นตัวอย่าง ต่อไปจะเป็นการอธิบายถึงการทำงานของอเวย์ทรีดีในขั้นพื้นฐาน

4.1.1.2 เริ่มต้นการเขียนโปรแกรม

หลังจากที่มีการทำงานร่วมกับอเวย์ทรีดีจะสังเกตเห็นว่าโปรเจคส่วนใหญ่มีวิธีการเริ่มต้นที่เหมือนกัน วัตถุพื้นฐานในอเวย์ทรีดี มีการสร้างขึ้น รวมทั้งวัตถุในมุมมอง 3 มิติโดยปกติจะอยู่ในคลาสด็อคคิวเม้น (document class) จุดเริ่มต้นหลักสำหรับโปรแกรมที่จะเขียนโค้ด ใน Flash Professional วิธีการเขียนคลาสด็อคคิวเม้นจะถูกใช้อยู่ตลอด เช่นเดียวกับกราฟฟิกทั้งหมดใน Flash และอเวย์ทรีดีต้องการที่จะให้ภาพมาจากวัตถุที่มีอยู่ในโครงสร้าง ซึ่งแตกต่างจากรายการที่แสดงโดยพื้นฐานในแฟลชการแสดงผลจะต้องสร้างทริกเกอร์ด้วยตนเอง และมีการจัดการมุมมองแสดง

ในอเวย์ทรีดี โดยคลาส View3D ในโปรเจคอเวย์ทรีดีมุมมองจะเป็นจุดเริ่มต้นและการประยุกต์ต้องใช้โค้ดพื้นฐานดังตัวอย่าง 4.2

ตัวอย่าง 4.2

```
package
    {
    import away3d.containers.View3D;
    import flash.display.Sprite;
    import flash.events.Event;
     public class MyFirstApp extends Sprite
        {
        private var view : View3D;
        public function MyFirstApp()
                {
                view = new View3D();
                view.x = 275;
                view.y = 200;
                addChild(_view);
                addEventListener(Event.ENTER_FRAME, onEnterFrame);
                }
        private function onEnterFrame(ev : Event) : void
                {
                view.render();
                }
        }
}
```

ในโค้ดก่อนหน้านี้มุมมองจะถูกสร้างขึ้น วางตำแหน่ง และบันทึกอยู่ในบรรทัดที่ 13-16 ซึ่งเป็นตัวหนาของคลาส View3D ครอบคลุม Sprite ซึ่งช่วยให้สามารถเพิ่มในรายการของการแสดงภาพในแฟลชเพื่อให้แสดงข้อมูลที่สามารถมองเห็น และในตำแหน่ง (x, y) ของมุมมองนี้จะแสดงถึงจุดลับสายตา ซึ่งมุมมอง 3 มิติหมายถึงตำแหน่งที่มุมมองของภาพที่ซึ่งปรากฏว่าเส้นขนานมาบรรจบกัน เพื่อสร้างภาพ 3 มิติที่ดูเหมือนจริง สำหรับจุดลับสายตาจะวางตำแหน่งที่ศูนย์กลางของแฟลชและต่อมาจะถูกสร้างขึ้นสำหรับการแสดงผลมุมมองในแต่ละเฟรม การแสดงผลจะเกิดขึ้นในบรรทัดที่ 22 ภายใต้คำสั่งของโค้ดที่เป็นตัวหนา

เพื่อคอมไพล์ไฟล์ MyFirstApp ไฟล์ตัวอย่างซึ่งเป็น SWF มีความจำเป็นต้องสร้างไฟล์คลาสภาษาแอกชันสคิปท์(ไฟล์ที่มีชื่อ MyFirstApp และนามสกุล. as) และวางลงในโค้ดข้างต้นผู้ใช้ Flash Professional ต้องวางไฟล์นี้ในคลาสซึ่งเป็นเส้นทางของ FLA ที่มีการใช้งาน (ซึ่งก็คือสถานที่ตั้งของไดเรกทอรีเดียวกัน) และให้แน่ใจว่าชื่อคลาสเป็นตามที่ระบุไว้ในคลาสด็อคคิวเม้นใน FLA เครื่องมืออื่นๆ ของภาษาแอกชันสคิปท์ (Flash Builder, FDT, FlashDevelop ฯลฯ) จะเรียบเรียงเป็น SWF โดยใช้คลาสด็อคคิวเม้นตามค่าเริ่มต้นและต้องการเพียงค่าที่ระบุไว้ในโปรเจคก่อนที่จะทำการคอมไพล์

การคอมไพล์ MyFirstApp จะพบว่าทั้งหมดที่แสดงเป็นหน้าต่างเปล่า นี้คือผลลัพธ์ที่คาดหวังไว้ จึงไม่ต้องกังวล เมื่อทำสำเร็จแล้ว สามารถเริ่มต้นการเพิ่มวัตถุ 3 มิติในการแสดงผล

4.1.2 การเพิ่มวัตถุ 3 มิติ ในฉาก

หน้าต่างหลักที่ใช้แสดงวัตถุ 3D เป็นที่รู้จักกันโดยทั่วไปว่าฉาก จะถูกแสดงโดยคลาส Scene3D ในอเว์ยทรีดีการทำงานกับฉากจะใกล้เคียงกับวิธีการใช้ เมื่อทำงานจะแสดงบนแฟลชในภาษาแอกชันสคิปท์ โดยใช้คำสั่ง เช่น addChild() และ removeChild() เมื่อมุมมองถูกสร้างในตัวอย่าง MyFirstApp ฉากเปล่าจะถูกสร้างขึ้นมุมมองตามค่าเริ่มต้น ซึ่งสามารถเข้าถึงได้ผ่านคุณสมบัติฉากในตัวอย่างคลาส View3D ตัวอย่างเช่น การสร้างลูกบาศก์ และเพิ่มลงในฉากสามารถทำได้โดยการเพิ่มโค้ดต่อไปนี้ที่ไฟล์ MyFirstApp ดังตัวอย่าง 4.3

ตัวอย่าง 4.3

```
var cube : Cube = new Cube();
_view.scene.addChild(cube);
```

ลูกบาศก์เป็นตัวอย่างของวัตถุพื้นฐาน 3 มิติและจำเป็นต้องมีแพคเกจที่อิมพอร์ตมาจาก away3d.primitives เพิ่มเข้าไป เพื่อให้แน่ใจว่าโค้ดจะคอมไพล์โดยไม่มีข้อผิดพลาดดังตัวอย่าง 4.4

ตัวอย่าง 4.4

```
import away3d.primitives.Cube;
```

จากตัวอย่างนี้ ภาพที่ปรากฏจะแสดงเป็นสี่เหลี่ยมซึ่งสร้างมาจากภาพสามเหลี่ยมสองรูป ลูกบาศก์จะมีสีทึบและเส้นขอบเป็นสีดำ คล้ายกับหลายๆเครื่องมือการสร้างภาพ 3 มิติ ที่วัตถุโดยส่วนใหญ่จะสร้างมาจากภาพสามเหลี่ยมที่รวมกัน สีที่ใช้สำหรับการแสดงผลลัพธ์ของลูกบาศก์มีรูปแบบจากค่าเริ่มต้น คือ WireColorMaterial ซึ่งมีประโยชน์ในการแก้จุดบกพร่องตามที่แสดงให้เห็นชัดเจนทั้งขอบ และรูปสามเหลี่ยมที่สร้างขึ้นเป็นลูกบาศก์พื้นฐาน ดังตัวอย่าง 4.5

ตัวอย่าง 4.5

```
var bmp : BitmapData = new BitmapData(200,200);
bmp.perlinNoise(200, 200, 2, Math.random(), true, true);
var mat : BitmapMaterial = new BitmapMaterial(bmp);
var cube : Cube= new Cube({ material: mat });
_view.scene.addChild(cube);
```

away3d.materials และภาพพื้นฐานของคลาส BitmapData ซึ่งอยู่ในแพคเกจ flash.display. จะเพิ่มคำสั่งเข้าไปยังล่างสุดต่อจากโค้ดเดิม ดังตัวอย่าง 4.6

ตัวอย่าง 4.6

```
import away3d.materials.BitmapMaterial;
import flash.display.BitmapData;
```

เมื่อคอมไพล์ผ่าน จะได้ลูกบาศก์ที่เหมือนก่อนหน้านี้ แต่มีสิ่งที่แตกต่าง คือวัตถุมีพื้นผิวหลากสี

รูป 4.2 ด้านหน้าของลูกบาศก์ ที่ได้จากการคอมไพล์

4.1.2.1 ทำความเข้าใจโครงสร้างและเริ่มต้นวัตถุ

โดยส่วนใหญ่วัตถุที่แท้จริงสามารถผ่านเข้าสู่ตัวสร้างวัตถุในอเวย์ทรีดีและจะเรียกว่าการเริ่มต้นวัตถุ (init สำหรับวัตถุที่ข้อมูลน้อย) ซึ่งทำงานรวดเร็ว และมีขนาดเล็ก ในการตั้งค่าเริ่มต้นของวัตถุที่ถูกสร้างขึ้น การใช้ init กับวัตถุเป็นวิธีของการทำงานสำหรับผู้ใช้ที่มีประสบการณ์ ซึ่งใช้กับข้อมูลที่มีขนาดเล็ก และจำนวนบรรทัดคำสั่งซึ่งใช้ตั่งค่าในอเวย์ทรีดีใช้คำสั่งที่น้อยกว่า เพื่อให้ได้ลักษณะวัตถุที่ต้องการ อย่างไรก็ตามวิธีการนี้ไม่สมควรใช้กับผู้เริ่มต้น เพราะว่าลักษณะที่ไม่ตายตัวของวัตถุ คุณลักษณะนี้จะเป็นประโยชน์ในการแก้ไขภาษาแอกชันสคิปท์เช่น เมื่อเขียนโค้ดเสร็จสิ้นไป ซึ่งอาจต้องการให้มีการเปิดใช้งาน เมื่อได้รับไลบรารีเข้ามา

วัตถุใดๆจะตั้งค่าเป็น init นอกจากนี้ยังสามารถตั้งค่าวัตถุให้เป็น public ภายหลังได้ (เช่น cube.material) ด้วยเหตุนี้โค้ดดังกล่าวจะเป็นการสร้างลูกบาศก์ที่เหมือนกับผลลัพธ์ของการใช้ init ในขณะเดียวกันก็ช่วยให้ ภาษาแอกชันสคิปท์มีโค้ดที่สมบูรณ์ ดังตัวอย่าง 4.7

ตัวอย่าง 4.7

```
var cube : Cube = new Cube();
cube.material = mat;
```

การเลือกใช้วัตถุที่เป็น init ในที่นี้ขึ้นอยู่กับตัวผู้ใช้เอง การตัดสินใจนี้เป็นการตัดสินใจส่วนตัว เนื่องจากข้อได้เปรียบบางประการ จากความช่วยเหลือ คือเมื่อมีการสร้างไลบรารีใหม่ หากตั้งค่าเป็น public นั้น จะสามารถติดต่อกับส่วนที่เหลือในโปรเจคได้ง่ายกว่า

4.1.3 การปรับแสงในฉาก

ใช้วิธีการเดียวกันกับการสร้าง 3 มิติเป็นลูกบาศก์ซึ่งอธิบายไว้ก่อนหน้านี้ โดยจะทำการสร้างทรงกลมเข้าไปในฉาก และสร้างบรรยากาศโดยเพิ่มแหล่งกำเนิดแสง พื้นผิวของทรงกลมจะเป็น Bitmap เช่นเดียวกัน มีโค้ดดังตัวอย่าง 4.8

ตัวอย่าง 4.8

```
var sphere : Sphere = new Sphere();
sphere.segmentsW = 32;
sphere.segmentsH = 32;
sphere.material = new BitmapMaterial(bmp);
sphere.x = 200;
_view.scene.addChild(sphere);
```

เช่นเดียวกับรูปลูกบาศก์ รูปทรงกลมนั้น จะต้องนำเข้าคลาสมาจากแพคเกจพื้นฐานด้วยการเพิ่มคำสั่งเข้ามา ดังตัวอย่าง 4.9

ตัวอย่าง 4.9

```
import away3d.primitives.Sphere;
```

ทรงกลมจะวางที่ตำแหน่งถัดไปอีก 200 หน่วย ในทางขวาของลูกบาศก์ และใช้พื้นผิวเช่นเดียวกัน เมื่อคอมไพล์อีกครั้งจะปรากฏรูปดังกล่าว

จุดนี้อาจมีข้อสงสัยว่าวัตถุมีช่วงที่เหมาะสมอย่างไรสำหรับภาพ 3D โดยวงกลม และสี่เหลี่ยมจะค่อนข้างเป็นไปในทางสองมิติ ทั้งนี้เนื่องจากความแตกต่างระหว่างรูปทรงกลม และวงกลม 2 มิติบนหน้าจอ (จอแสดงผลที่ใช้บนแฟลช) คือการรับรู้เชิงลึก จำเป็นต้องหลอกสายตา ว่าพวกเขากำลังดูวัตถุ 3 มิติอยู่ และวิธีหนึ่งในการทำขึ้นอยู่กับแสงและเงา หากแหล่งกำเนิดแสงเป็นรูปทรงกลมอยู่เหนือภาพ จึงเป็นไปได้ที่จะเห็นความลึกของภาพจากแสงและเงาที่ใช้กับวัตถุ

ในอเวย์ทรีดีการเลือกใช้พื้นผิวที่จะกำหนดการแรเงาจากแหล่งกำเนิดแสงใดๆ สามารถแสดงผลได้ ในตัวอย่าง BitmapMaterial จะใช้กับทั้งลูกบาศก์ และทรงกลมซึ่งไม่มีคุณสมบัติของแรเงา ดังนั้นจึงไม่ขึ้นอยู่กับแหล่งกำเนิดแสง เมื่อต้องการใช้การแรเงา จึงจำเป็นต้องเลือกพื้นผิวที่สนันสนุนการทำงานนี้ วิธีที่ง่ายที่สุดคือการใช้คลาส WhiteShadingBitmapMaterial โดยทำการเปลี่ยนคำสั่งในบรรทัดที่ 4 ซึ่งเป็นตัวหนา จะเป็นการตั้งค่าพื้นผิวของทรงกลม สามารถใช้งานแสงและเงาโดยการเปลี่ยนโค้ดต่อไปนี้ลงไปแทนที่โค้ดเดิม และต้องไม่ลืมที่จะนำเข้าคลาส WhiteShadingBitmapMaterial จากแพคเกจ away3d.materials ดังตัวอย่าง 4.10

ตัวอย่าง 4.10

```
sphere.material = new WhiteShadingBitmapMaterial(bmp);
```

ทำการคอมไพล์อีกครั้ง จะเห็นได้ว่าทรงกลมจะดูมีมิติมากขึ้น และกลายเป็นรูปทรง 3 มิติ สีทั้งหมดจะหายไป และแทนที่ด้วยด้วยสีดำภายในฉาก หากไม่มีแหล่งกำเนิดแสงในฉาก เพื่อให้ปรากฏภาพวัตถุที่มีการแรเงา จะทำให้แสงและเงานั้นต้องแสดงในความมืด การเพิ่มแหล่งกำเนิดแสงจะแก้ปัญหานี้ โดยโค้ดดังตัวอย่าง 4.11

ตัวอย่าง 4.11

```
var light : PointLight3D = new PointLight3D();
light.y = 500;
_view.scene.addLight(light);
```

ในบรรทัดที่ 1 ของโค้ดข้างต้นจะสร้างวัตถุ PointLight3D ขึ้นใหม่ (ซึ่งเป็นคลาสที่นำเข้าจากแพคเกจ away3d.lights) และเป็นจุดเดียวที่แสงที่ปล่อยออกมาในทุกทิศทาง ในบรรทัดที่ 2 จะย้ายตำแหน่งของแหล่งกำเนิดแสงไป 500 หน่วยขึ้นไปด้านบน และในบรรทัดที่ 3 เพิ่มวัตถุเข้าไปยังฉาก เมื่อคอมไพล์ไฟล์ MyFirstApp ทรงกลมจะมีแสงและเงาบนพื้นผิวในภาพ 3 มิติ

รูป 4.3 ภาพทรงกลมที่มีแสงและเงาปรากฏอยู่ด้านบน

4.1.4 ภาพเคลื่อนไหว 3 มิติ

จนถึงตอนนี้ การแสดงภาพของทุกๆเฟรมจะเปลี่ยนไปเพียงเล็กน้อย ไม่มีเหตุผลที่จะให้ CPU ทำงานเพื่อคำนวณและแสดงผล 3 มิติในฉากถึง 25 ครั้ง เมื่อแต่ละเฟรมมีลักษณะเหมือนกัน หากเป็นเช่นนี้ก็ไม่จำเป็นต้องทำภาพเคลื่อนไหว

ระบบแคชอัตโนมัติในอเว่ย์ทรีดีเพื่อให้แน่ใจว่าการแสดงผลสามารถดูได้ซ้ำๆ ซึ่งฉากจะไม่มีการเปลี่ยนแปลงทำให้ CPU ไม่ต้องทำงานหนักในแต่ละครั้งที่แสดงผล จะมีก็แต่พื้นที่ภายในมุมมองเท่านั้นที่มีการเปลี่ยนแปลง ซึ่งเรียกว่าการอัฟเดท

การเพิ่มปริมาณการหมุนของภาพทรงกลมในฉากแต่ละเฟรม โดยสามารถทำการหมุนวัตถุให้เลียนแบบตามแกนของตัวเอง แต่ก่อนที่จะทำเช่นนั้น การอ้างอิงถึงทรงกลมจะต้องถูกเก็บไว้ในคลาสด็อคคิวเม้น เพื่อให้ทรงกลมสามารถเคลื่อนไหวใน onEnterFrame เพื่อให้สำเร็จตามนี้ ต้องย้ายตัวแปรของทรงกลม ประกาศขอบเขตของคลาส และทำให้มันเป็นตัวแปรแบบ private นี่คือข้อตกลงที่ใช้บ่อยในการเขียนโปรแกรมเชิงวัตถุสำหรับตัวแปรแบบคลาส private และจะเป็นเหมือนตัวอย่างโค้ดดังตัวอย่าง 4.12

ตัวอย่าง 4.12

```
public class MyFirstApp extends Sprite
{ private var sphere : _Sphere;
...
```

ในฟังก์ชั่นจัดการ onEnterFrame จะเพิ่มการหมุนตามแกนในแนวตั้ง (Y) การหมุนแกนจะหมุนทรงกลมในลักษณะที่ต้องการ ดังตัวอย่าง 4.13

ตัวอย่าง 4.13

```
private function onEnterFrame(ev : Event) : void
{       _sphere.rotationY += 5;
        view.render(); }
```

ทำการคอมไพล์อีกครั้ง ตัวอย่าง MyFirstApp จะแสดงรูปทรงกลมหมุนรอบแกน Y เทคนิคภาพเคลื่อนไหวนี้เป็นตัวอย่างที่ง่ายใน 3 มิติ และในอเว์ยทรีดีมีวิธีการหลากหลายในการสร้างภาพเคลื่อนไหว

4.1.5 การติดต่อสื่อสารใน 3 มิติ

การสร้างปุ่ม รูปแบบพื้นฐาน โดยส่วนใหญ่ของการติดต่อนั้น จะทำโดยเมาส์ ซึ่งเป็นเรื่องง่าย เพื่อให้สามารถทำงานในอเว์ยทรีดีที่อยู่ในแฟลชซึ่งเกี่ยวข้องกับการเพิ่มกิจกรรมเพื่อวัตถุ 3 มิติที่ตอบสนองต่อเหตุการณ์พิเศษโดยใช้ MouseEvent3D คลาส MouseEvent3D จะอยู่ร่วมกับเหตุการณ์แบบกำหนดเองสำหรับอเว์ยทรีดีเพื่ออธิบายวิธีการพื้นฐานจะตั้งค่าเพื่อให้โปรแกรมที่รอรับคำสั่งสร้างลูกบาศก์ในฉาก เมื่อตรวจพบการคลิกเมาส์ ในขั้นตอนแรกเป้าหมายนี้คือการสร้างฟังก์ชั่นจัดการเหตุการณ์ ดังตัวอย่าง 4.14

ตัวอย่าง 4.14

```
private function onClickCube(ev : MouseEvent3D) : void
{       GenericTweener.tween(ev.currentTarget, 1);
        {       rotationX: Math.random()*360,
                rotationY: Math.random()*360,
                rotationZ: Math.random()*360 } }
```

ในโค้ดที่ไว้คอยดักจับเหตุการณ์ MouseEvent3D เช่นการคลิกเมาส์ จะใช้คลาสขนาดเล็ก tweening ที่เรียกว่า GenericTweener เพื่อให้การหมุนเวียนทั้งหมดของลูกบาศก์ราบรื่นตามแกนโดยสุ่มค่า คลาส GenericTweener เป็นส่วนหนึ่งของแพคเกจไฟล์ และจะอยู่ในแพคเกจ flash3dbook.common จะใช้งานได้ต่อไป การใช้ addEventListener() โดยทั่วไป วิธีการที่ทำบนวัตถุอย่างฟังก์ชั่น onClickCube สามารถกำหนดเป็นตัวจัดการสำหรับเหตุการณ์ MouseEvent3D.MOUSE_UP ดังตัวอย่าง 4.15

ตัวอย่าง 4.15

```
cube.addEventListener(MouseEvent3D.MOUSE_UP, onClickCube);
```

แทรกโค้ดบรรทัดนี้ ใต้โค้ดที่เกี่ยวข้องกับการสร้างลูกบาศก์ จะเสร็จสิ้นขั้นตอนของการเพิ่มการติดต่อ หากคอมไพล์ MyFirstApp และลองคลิกที่ลูกบาศก์ที่จะเกิดการหมุน

4.2 มุมมอง ฉาก และกล้อง

ที่ผ่านมาเป็นการสร้างโปรเจคในอเวย์ทรีดีโดยเพิ่ม View3D ใหม่ลงในโปรเจค โดยทั่วไปสภาพแวดล้อมพื้นฐานของ 3 มิติจะประกอบด้วยมุมมอง ฉาก และกล้องถ่ายรูป

4.2.1 การทำความเข้าใจเบื้องต้น

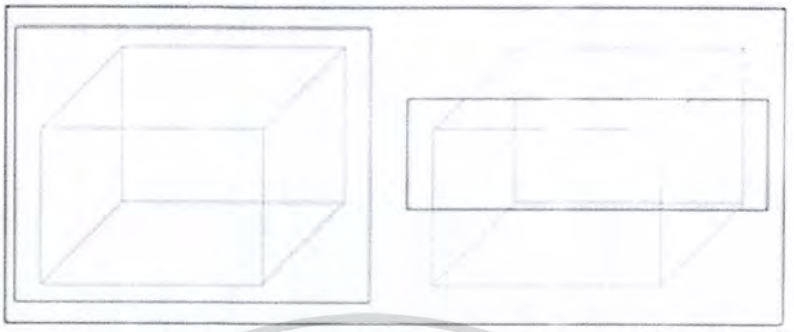
สิ่งที่สำคัญจะต้องมีความรู้พื้นฐานเกี่ยวกับกฎของมุมมอง ฉาก และกล้องถ่ายรูปใน 3 มิติ ก่อนที่จะทำการตั้งค่าต่างๆ ในอเวย์ทรีดีทั้งมุมมอง ฉาก และการตั้งค่าของกล้องถ่ายรูป

4.2.1.1 มุมมอง

มุมมองถือเป็นตัวแทนของภาพ 2 มิติ ให้เป็น 3 มิติ ที่มีการแสดงผลและการเชื่อมโยงโลก 3 มิติเสมือนจริงของโลก 2 มิติที่มีความจำเป็นสำหรับการแสดงบนหน้าจอคอมพิวเตอร์ และจะมีสิ่งที่น่าสนใจคือวิวพอร์ตซึ่งหมายถึงพื้นที่รูปสี่เหลี่ยมที่มีผลลัพธ์กับการมองเห็นของฉาก 3D viewport คือส่วนแสดงผลจากสิ่งที่กล้องสามารถมองเห็น ในการรวมภาพของวัตถุจะถูกแสดงโดยผ่านเลนส์ของกล้อง เลนส์เป็นหน้าต่างที่แสดงให้เห็นในฉากสามมิติ ซึ่งสามารถทำให้หน้าต่างนี้เล็กและเห็นเพียงส่วนเล็กๆ ของฉากหรือทำให้มันใหญ่และสามารถเห็นได้มากขึ้น ส่วนนี้ถ้าให้เปรียบเทียบก็คล้ายกับหน้าต่างจริงๆที่ผนังบ้าน การทำให้หน้าต่างใหญ่จะไม่มีผลต่อโลกภายนอก แต่จะมีผลต่อผู้ที่สังเกตุอยู่ ที่สามารถดูโลกภายนอกจากกรอบของหน้าต่างนี้

ภาพประกอบต่อไปนี้ ทางด้านซ้ายของภาพจะมีวิวพอร์ตค่อนข้างใหญ่และทางด้านขวาวิวพอร์ตเล็กและกว้าง กรอบมืดหนาแสดงเค้าร่างวิวพอร์ตในภาพขวาจะเห็นอย่าง

ชัดเจนว่าหน้าต่างในโลกสามมิติขนาดและวิธีการเปลี่ยนแปลงขนาดของมันจะไม่ยึดขนาดของวัตถุ 3 มิติในฉากเลย แต่มันจะมีผลต่อวิสัยทัศน์ในเห็นวัตถุ



รูป 4.4 วิวพอร์ตที่จะไม่เปลี่ยนแปลงตามขนาดของวัตถุ

4.2.1.2 ฉาก

ฉากเป็นองค์ประกอบทั้งหมดของวัตถุที่จะปรากฏออกมาในพื้นที่ฉาก 3 มิติ เป็นขั้นตอนในแฟลชกับทั้งสามแกน คือ x y และ z วัตถุที่ต้องการจะให้แสดงผลในแต่ละชิ้น จะต้องเพิ่มเข้าไปให้อยู่ในฉาก ถ้าไม่ใส่วัตถุเข้าไปในฉากวัตถุเหล่านั้นจะไม่สามารถปรากฏบนหน้าจอได้

ฉากในโปรแกรมอเวย์ทรีดีจะแสดงผลเป็นคลาส Scene3D ตั้งอยู่ในแพคเกจ away3d.containers จะแสดงผลวัตถุเป็น 3 มิติในแฟลชวัตถุที่รวมอยู่ในการแสดงผลที่แนบมาภายในคลาส Scene3D

จาก Flash 10 เป็นต้นไป วัตถุพื้นฐานในแฟลชจะมีลักษณะเป็น 3 มิติ เกือบเหมือนธรรมชาติ ในอเวย์ทรีดีนี้จะเป็นการแสดงวัตถุทั้งหมดที่สามารถปรากฏในฉาก (Sprite, MovieClip, ฯลฯ) ซึ่งมีแกน x y และ z เป็นคุณสมบัติของตำแหน่งในการหมุนและการปรับขนาด

เช่นเดียวกับคลาสในแฟลชจะสืบทอดจาก flash.display คลาส DisplayObject ผลิตวัตถุในกรณีที่สามารถวางบนฉาก ในอเวย์ทรีดีที่สืบทอดจากคลาส away3d.core.base และ Object3D ผลิตวัตถุที่สามารถวางในฉาก

กรณีที่มีการแสดงรายการพื้นฐานในแฟลชการแสดงผล 3 มิติจะถูกสร้างขึ้นจากลำดับชั้น ซึ่งวัตถุ 3 มิติสามารถซ้อนกันอยู่ภายในระยะทางระหว่างวัตถุกับฉาก วัตถุ 3 มิติ เหล่านี้จะมองเห็นเมื่อแสดงผลและแสดงจริงใน (แสดงเป็นคลาส Scene3D) เป็น subclass ของฉาก 3 มิติ คลาส ObjectContainer3D จะกล่าวถึงวิธีการใช้ฉาก ประโยชน์ของการวางตำแหน่ง และภาพเคลื่อนไหวของวัตถุ 3 มิติ

4.2.1.3 กล้อง

สามารถสมมุติได้ว่ากล้องในที่นี้ เป็นกล้องจริงๆ ซึ่งอยู่ในพื้นที่ฉาก 3 มิติ กล้องจะกำหนดมุมมองที่จะปรากฏออกมาทางหน้าจอแสดงผล เพราะกล้องในสามมิติเป็นวัตถุที่มองไม่เห็นจึงไม่เป็นส่วนหนึ่งในฉาก และไม่จำเป็นต้องทำอะไรเพิ่มเติม

กล้องในพื้นที่สามมิติจะทำงานได้มากกว่ากล้องจริง ตัวอย่างเช่นในระบบสามมิติสามารถแยกวัตถุที่จะแสดงผลเมื่อมีวัตถุที่ใกล้เกินไปหรือไกลจากกล้องเกินไป กล้องสามารถละวัตถุนั้นที่ไม่ได้อยู่ในช่วงที่สามารถมองเห็น ด้วยเหตุผลทางด้านประสิทธิภาพของวัตถุที่อยู่ไกลหรือแม้กระทั่งวัตถุที่อยู่หลังกล้องไม่จำเป็นต้องบันทึกภาพ เพื่อเป็นการประหยัดการคำนวณวัตถุที่อยู่ในตำแหน่งที่กล้องไม่สามารถมองเห็น

วัตถุ Camera3D สืบทอดคลาสจาก Object3D หมายถึงจุดของการสังเกตภายในฉาก มีอิทธิพลต่อสิ่งที่จะมองเห็นในมุมมอง โดยการคำนวณภาพที่ฉายบนมุมมองของตำแหน่งกล้องและการหมุน คล้ายกับว่ามีการแสดงเนื้อหาของฉากจากมุมมองของกล้อง Camera3D

กล้องถ่ายรูปนั้นวัตถุ 3 มิติในพิกัดฉากจะเปลี่ยนเป็นภาพ 2 มิติในพิกัดมุมมองนอกเหนือจากตำแหน่งและการหมุน กล้องถ่ายรูปสามารถปรับเปลี่ยนฉาก มีการแสดง เช่น ซูม โฟกัส และคุณสมบัติเลนส์ กำหนดประเภทของการประมาณผล คุณลักษณะเหล่านี้ส่งผลกระทบต่อผลที่ให้ในลักษณะเดียวกับที่การเลือกเลนส์ที่แตกต่างกัน และการตั้งค่าสำหรับกล้องจริงมีผลต่อการถ่ายภาพ

4.2.2 การทำงานพื้นฐานของ 3 มิติ

อเว่ย์ทรีดีใช้หลักการสำคัญด้วย perspective projection engine ซึ่งเป็น projection มีความหมาย คือการแปลงวัตถุสามมิติในสเปซให้กลายเป็นภาพสองมิติตามแกน x และ y ซึ่งมีสมการปรับมุมมอง(perspective scaling equation) ดังนี้ ดังสมการ 4.1

$$T = scale = focus\ length / (focus\ length + z) \qquad (4.1)$$

ในสมการข้างต้น T เป็นขนาดของมุมมอง (perspective scale) และ z เป็นระยะห่างของ projection plane ส่วนระยะ focus length หรือเรียกว่า screen location ขึ้นอยู่กับจำนวนของ perspective ใน view โดยใช้สมการในการสร้างภาพ สามมิติเสมือนบนจอภาพสองมิติที่มีความลึกและ scaling

4.2.2.1 การทำงานของพิกัด 3 มิติในพื้นที่

จะคล้ายกับระบบพิกัด 2 มิติในหลายๆตัว หากไม่ได้ศึกษาพวกมันในหลักสูตรพีชคณิต หรืออาจจะได้ทำงานกับพวกมัน ใน flash Photoshop หรือแม้กระทั่งซอฟต์แวร์ต่างๆ

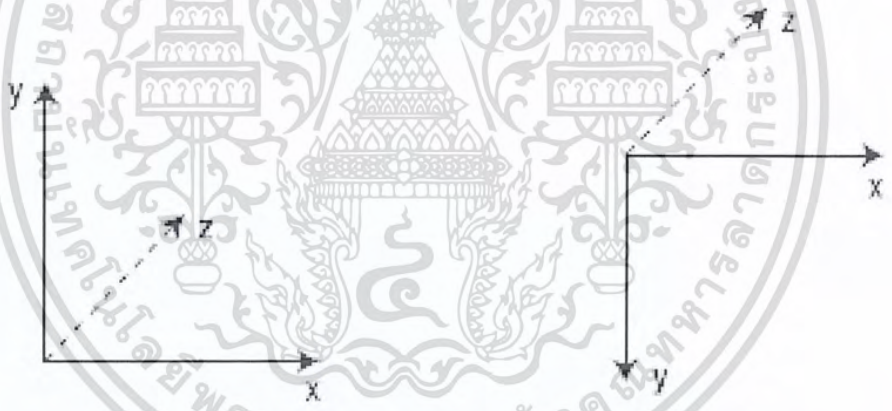
ระบบพิกัดนั้นเป็นวิธีการกำหนดตำแหน่งในพื้นที่ โดยระยะทางจากศูนย์ บอกเป็นจำนวนระยะทาง แต่ละส่วนมีระยะทางทิศทางที่กำหนดโดยแกน ซึ่งเป็นจุดพิกัดในพื้นที่ โดยระยะทางเพิ่มขึ้นหรือลดในแต่ละทิศทาง

ในพื้นที่ 2D ส่วนใหญ่ใช้ระบบพิกัดสองทิศทางตั้งฉากกัน โดยมีแกน X และแกน Y สำหรับซอฟต์แวร์เหล่านี้ ตำแหน่งขององค์ประกอบกราฟิกบนหน้าจอสามารถกำหนดได้โดยใช้ x และ y กำหนดค่าพิกัดของแกนเหล่านี้

ค่าพิกัดมีแกนกำหนดเป็นระยะทางจากจุดศูนย์ ประเด็นนี้ได้โดยทั่วไป ตำแหน่งที่ทั้งสองแกนตัดกัน ซึ่งระยะทางในแต่ละแกนเป็นศูนย์เรียกว่าจุดเกิด

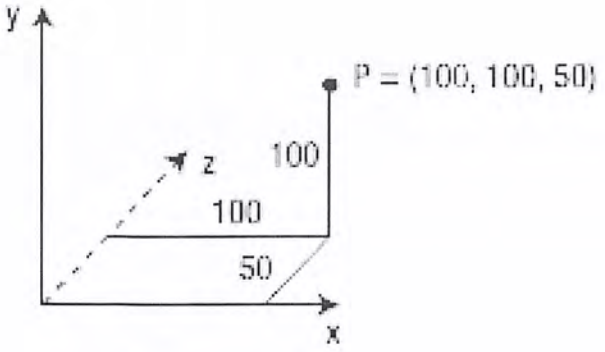
ในระบบ3 มิติมีพิกัดสำคัญที่เพิ่มมาจากระบบ 2D โดยมีแกนที่สามคือแกน Z ซึ่งอยู่ในตำแหน่งตั้งฉากกับอีกสองแกน เพื่อให้วัตถุสามารถเคลื่อนที่เข้า และออก จาก viewpoint โดยใช้ perspective scaling

ในอเวย์ทรีดีจะมีแกนหลักสามมิติ แตกต่างกับแฟลชอยู่เล็กน้อย ดังที่แสดงในภาพ ในอเวย์ทรีดีนั้น แกน Y ที่มีหัวลูกศรชี้ลง จะมีค่าเป็นบวก



รูป 4.5 อเวย์ทรีดีระบบพิกัด (ซ้าย) เมื่อเทียบกับระบบพิกัดพื้นฐาน Flash 10 (ขวา)

ระบบ 3 มิติ ใช้ระบบพิกัดตำแหน่งโดยกำหนดระยะทางแต่ละแกน ข้อกำหนดของตำแหน่งนี้เป็นบางครั้งเรียกว่าเวกเตอร์ของจุด ส่วนประกอบ X Y และค่า Z ซึ่งสอดคล้องกับระยะทางของจุด เวกเตอร์ดังกล่าวจะถูกเขียนโดยทั่วไป (X, Y, Z) ซึ่งก็คือค่าสามค่าตามลำดับตัวอักษรคั่นด้วยเครื่องหมายจุลภาคและล้อมรอบด้วยวงเล็บ ในรูปแสดงให้เห็นว่าจุดของเวกเตอร์จะคำนวณจากแกนของระบบการประสานงานใน 3D



รูป 4.6 จุด P ตำแหน่งที่ X = 100, Y = 100, Z = 50 ระบบพิกัดในอเวย์ทรีดี

4.2.2.2 ทฤษฎีของ Thales

การสร้าง 3 มิติ โดยใช้ Flash Engine ก่อนที่ flash 10 จะรองรับ 3d การเริ่มต้นที่จะสร้างภาพสามมิติ นั้น เริ่มจากการใช้แกน x และ y จากนั้นสร้างระบบแกน z เอง วิธีที่สร้างระบบแกน z จะใช้ perspective scaling ซึ่งหมายความว่า ถ้าวัตถุเคลื่อนที่ไกลออกไป วัตถุจะมีขนาดเล็กลง ถ้าวัตถุเคลื่อนที่ เข้ามาใกล้ วัตถุจะมีขนาดใหญ่ขึ้น ดังนั้นจะเริ่มเขียนโปรแกรมด้วยทฤษฎีของ Thales ว่าด้วยหลักการ สามเหลี่ยมคล้าย

Thales Theorem ชาวกรีกมีชื่อเสียงเป็นอย่างมากในด้านของตรีโกณมิติ และ Thales เป็นคน แรกที่คิดค้นทฤษฎีของสามเหลี่ยมคล้าย จาก concept ของสามเหลี่ยมคล้าย ทำให้สามารถสร้าง linear projection ซึ่งเป็นหัวใจของ flash 3d engine

ตัวอย่างการนำทฤษฎีมาประยุกต์ใช้ สมมติว่าคุณอยู่ในบ้าน มองหน้าต่าง ซึ่งเป็น object อย่าง หนึ่ง ถ้าคุณเดินออกมาไกลจากหน้าต่าง คุณจะมองเห็นวัตถุนอกหน้าต่างเล็กลง ถ้าคุณเดินเข้าใกล้ หน้าต่าง จะเห็นวัตถุนอกหน้าต่างใหญ่ขึ้น ระยะทางระหว่างหน้าต่างกับตัวคุณเรียกว่า focus length ซึ่งในที่นี้หน้าต่างเป็น projection plane (หรือวิวพอร์ต) ตาของคุณเป็น vanishing point

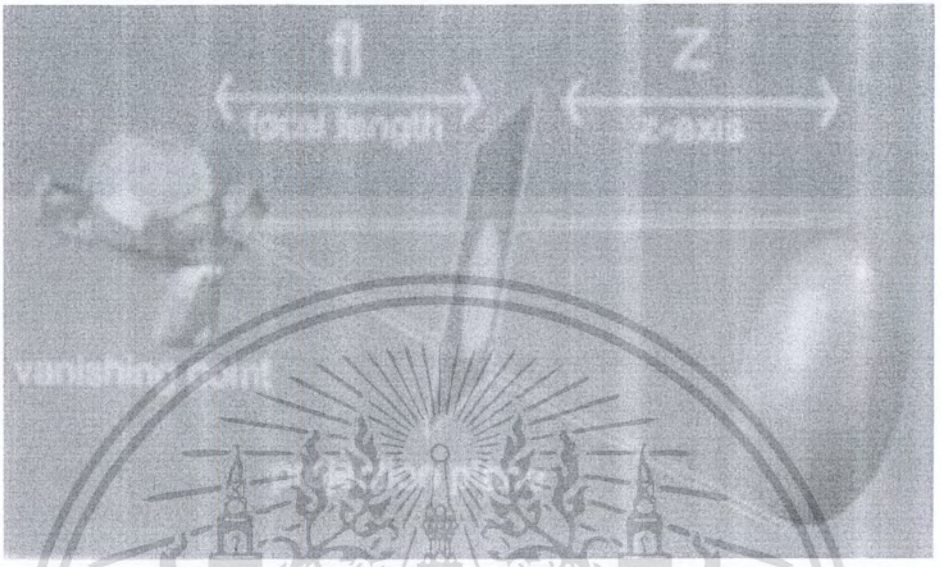
ถ้าลอง fixes focus length คือไม่เดินเข้าใกล้ และเดินออกจากหน้าต่าง มองออกไปหาวัตถุ นอกหน้าต่าง เมื่อมีนกบินมาที่หน้าต่างนกจะมีขนาดใหญ่ขึ้น แต่เมื่อนกบินออกไปจากหน้าต่างขนาด ของนกก็จะเล็กลงไปเรื่อยๆ นี่คือตัวอย่างของ z-axis ระยะห่างระหว่างตัววัตถุนอกหน้าต่าง และ หน้าต่าง ซึ่งเป็นเสมือนวิวพอร์ตหรือหน้าจอคอมพิวเตอร์ที่แสดงผล

สมการทฤษฎี Thales เป็นดังสมการ 4.2

$$T = \text{scale} = \text{focus length} / (\text{focus length} + z) \qquad (4.2)$$

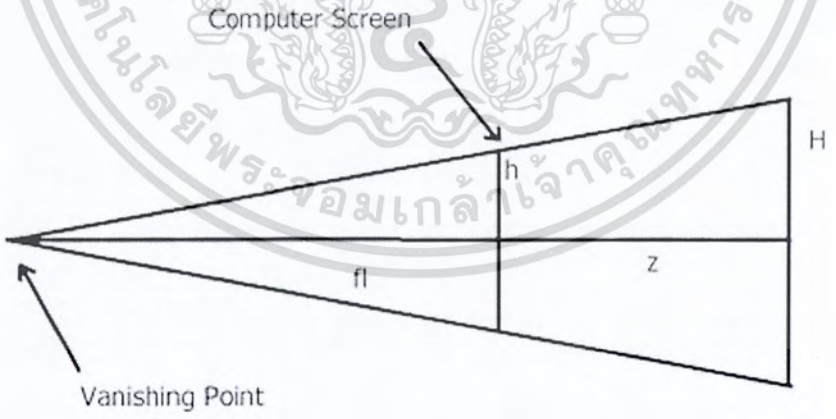
ถ้า T มีค่าเป็น 1 หมายความว่า วัตถุนอกหน้าต่างอยู่ติดกับหน้าต่างเลย

ถ้า T มีค่าเป็น 0 หมายความว่า วัตถุนอกหน้าต่างอยู่ไกลมากจนมองไม่เห็น



รูป 4.7 สมการทฤษฎี Thales

การนำทฤษฎีมาประยุกต์ใช้



รูป 4.8 สมการทฤษฎี Thales ตามสมการ 4.3

h เป็นขนาดของ object บนจอคอมพิวเตอร์

H เป็นขนาดของวัตถุขนาดจริง

fl เป็น focus length คือระยะห่างระหว่าง หน้าจอคอมพิวเตอร์จนถึงตาของ

z เป็นระยะห่างระหว่างจอและ object ที่แสดง

จากกฎของสามเหลี่ยมคล้าย จะได้สมการ 4.3

$$h/fl = H / (fl + z) \qquad (4.3)$$

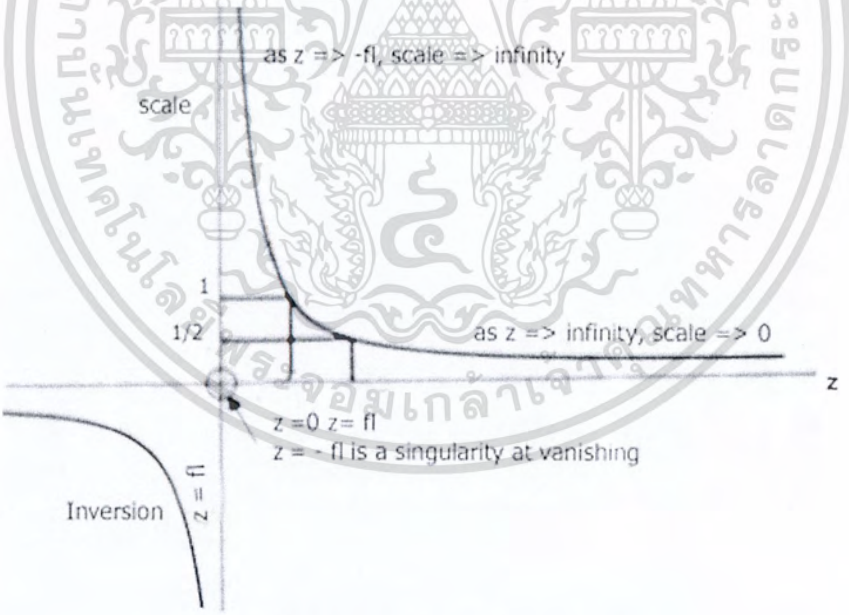
เมื่อนำมาทำการ cross-multiplying จะได้สมการ scaling equation

$$Scale = h/H = fl / (fl + z) \qquad (4.4)$$

ซึ่งแปลความหมายของสมการว่าขนาด Scale ของวัตถุที่แสดง จะเท่ากับ h / H คือ ขนาดที่ แสดงบนจอส่วนด้านขนาดของวัตถจริง และเท่ากับ focus length ส่วนด้วย focus length บวกกับระยะ ความลึก (z) นั่นเอง

ถ้า z เป็น 0 คือวัตถุอยู่ติดกับจอ scale = 1

ถ้า z = fl ทำให้ image scale เป็น 1/2



รูป 4.9 ความสัมพันธ์ของสมการ

จากภาพพบว่าเป็นกราฟที่คล้ายคลึงกับกราฟแรงโน้มถ่วง หรือ electrical potential curves ที่พบใน physics ซึ่งพบว่า size scaling สามารถสร้างจากการ differential ของ scaling equation ได้ด้วย

4.2.2.3 การแสดงผล

อเวย์ทรีดีจะมีลักษณะพื้นฐานดังต่อไปนี้ คือ การตัดการ เรียงลำดับ Z และประมาณการมุมมอง

4.2.2.3.1 Clipping

เนื่องจากภาพที่จะแสดงให้เห็นบนจอภาพนั้น อาจจะแสดงให้เห็นทั้งหมดของโลก 3D หรืออาจจะแค่มุมเล็ก ๆ เท่านั้น การทำ clipping คือการใส่กรอบว่า มุมมองที่ภาพจะปรากฏบนจอภาพนั้น มีขอบเขตเท่าไหร่ ทำให้ไม่จำเป็นต้องเสียเวลาไปกับการคำนวณภาพที่อยู่นอกเหนือมุมมองที่จะแสดงผล วิธีการง่ายๆในการระบุนี้ คือการพิจารณาตำแหน่งของเวกเตอร์เมื่อเทียบกับขอบของมุมมอง (ตามที่กำหนดขอบของวิวพอร์ตหรือขอบของแฟลช) ถ้าเวกเตอร์อยู่นอกมุมมองพื้นที่ ก็จะถูกตั้งค่าสถานะสำหรับการกำจัดออก ก่อนที่จะดำเนินการให้เกิดขึ้น

4.2.2.3.2 Z-sorting

ปัญหา z - sorting เป็นปัญหาที่พบบ่อยที่สุดทำให้ไม่ได้ภาพดังที่ต้องการ เป็นปัญหาที่เกิดจากการเรียงลำดับของวัตถุ เพื่อให้เข้าใจอย่างถ่องแท้ จะอธิบายถึงปัญหานี้อย่างละเอียด โดยยึดใน Papervision3D

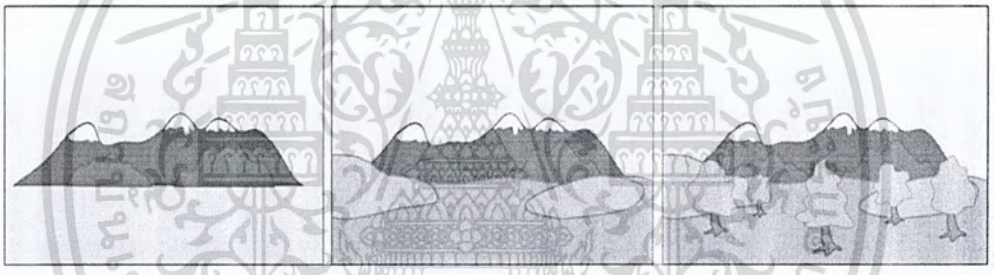
รูป 4.10 ปัญหา z – sorting

สังเกตว่าบางส่วนรูปสามเหลี่ยมดูเหมือนจะขาดหายไป สามเหลี่ยมที่ขาดหายไปของพวยกาน้ำซึ่งตัวพวยกาน้ำถูกเรียงลำดับการแสดงให้อยู่ด้านหลังสามเหลี่ยมของตัวกาน้ำ ปัญหานี้จึงเป็นหน้าที่หลักของงาน z - sorting ใน Papervision3D

1) The painter's algorithm ซึ่ง z - sorting เป็นกระบวนการกำหนดความลึกของรูปสามเหลี่ยมแต่ละรูปที่จะวาดบนหน้าจอ เป็นการกำหนดลำดับที่จะวาดรูปสามเหลี่ยมใดอยู่ด้านบนของสามเหลี่ยมส่วนอื่นๆตามระยะทางที่ห่างจากกล้อง CPU intensive(การ

คำนวณแบบต่อเนื่อง) เป็นกระบวนการพิจารณาที่สามเหลี่ยมควรจะอยู่ในหน้า เป็นวิธีที่รวดเร็ว แต่ไม่ถึงกับแม่นยำมากที่จะเรียงลำดับของรูปสามเหลี่ยมแต่ละรูป algorithm นี้ถูกเรียกกันว่า painter's algorithm เป็นชื่อที่ได้มาจากเทคนิคของจิตรกรที่ใช้ในการวาดภาพโดยจิตรกรมักวาดส่วนที่ไกลของฉากมาวาดก่อน

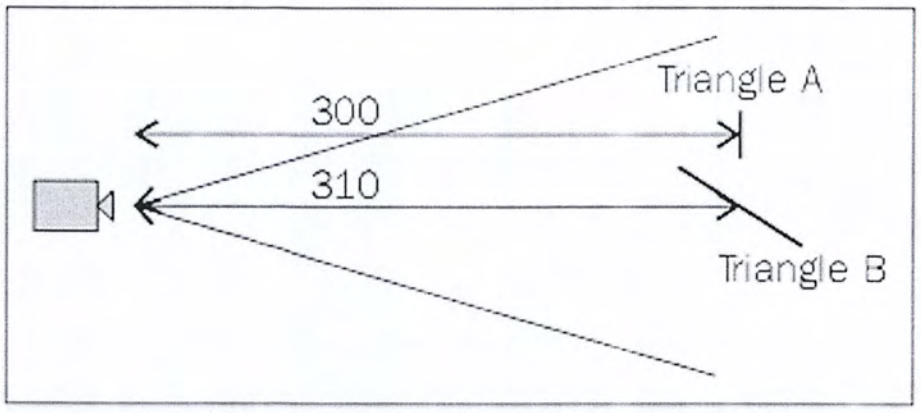
2) ลำดับการทำงานของ painter's algorithm จะแบ่งเป็นขั้นตอนต่างๆ ขั้นตอนแรก จัดลำดับของตำแหน่งสามเหลี่ยมแต่ละรูปที่อยู่ใน camera frustum ที่มี ตำแหน่งใน frustum เรียงจากตำแหน่งที่ไกลมากที่สุดไปถึงตำแหน่งที่ใกล้ที่สุดโดยมีความสัมพันธ์กับกล้อง ขั้นตอนต่อไปวาดสามเหลี่ยมบนหน้าจอตามลำดับการเรียงที่ทำไว้ วิธีนี้เป็นการเริ่มต้นด้วยการวาดภาพที่ไกลที่สุดก่อนและวาดภาพสุดท้ายด้วยภาพสามเหลี่ยมใกล้ที่สุด ดังภาพประกอบต่อไปนี้



รูป 4.11 ลำดับการทำงานของ painter's algorithm

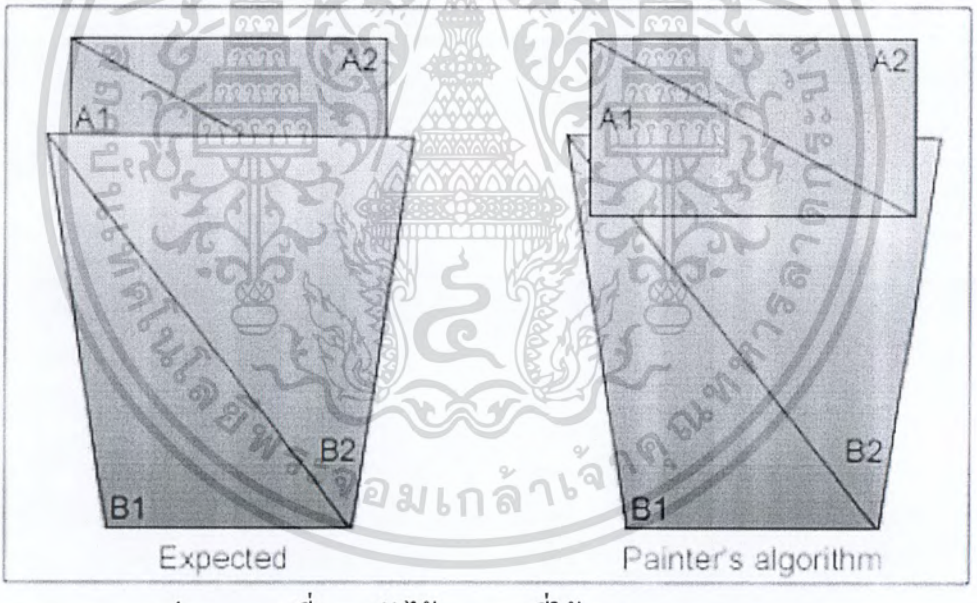
ภาพทางด้านซ้ายภูเขาเป็นสิ่งที่อยู่ไกลที่สุดจากกล้อง ภาพต่อมาตรงกลางหญ้าตามภูมิประเทศและภาพสุดท้ายทางด้านขวาเป็นต้นไม้เป็นสิ่งที่ใกล้ที่สุดกับกล้อง

3) การเรียงลำดับภาพสามเหลี่ยม(sorting triangles) เพื่อทำความเข้าใจปัญหาที่มีความเกี่ยวข้องกับ z - sorting จำเป็นต้องทราบถึงวิธีการที่สามเหลี่ยมจะถูกเรียงลำดับ ภาพต่อไปนี้แสดงตัวอย่างรูปสามเหลี่ยมสองภาพและระยะทางของภาพสามเหลี่ยมมาถึงกล้อง



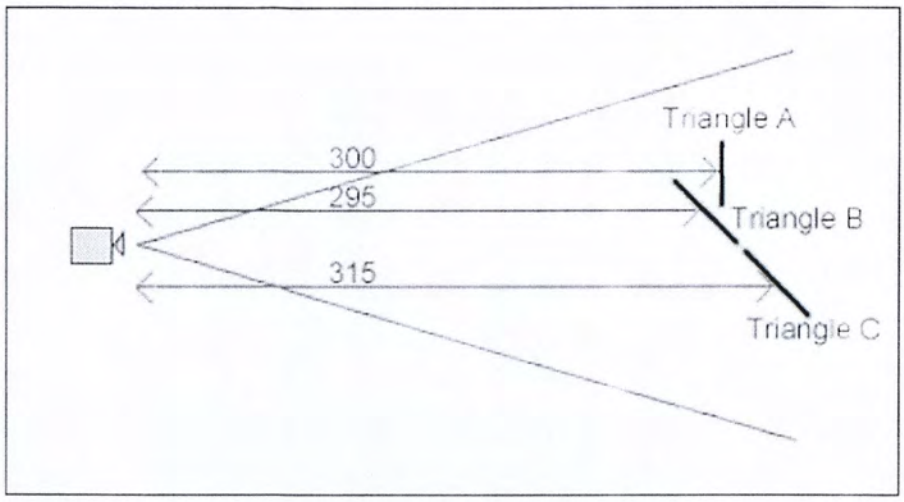
รูป 4.12 สามเหลี่ยมที่มีระยะทางของภาพแตกต่างกัน

การวาดรูปสามเหลี่ยมเหล่านี้ในลำดับที่ถูกต้อง ภาพสามเหลี่ยม A ควรถูกวาดก่อนภาพสามเหลี่ยม B แต่วิธีของจิตรกร(painter's algorithm)วาดภาพสามเหลี่ยมไกลที่สุดก่อนคือภาพสามเหลี่ยม B แล้วตามด้วยภาพสามเหลี่ยมใกล้ซึ่งเป็นภาพสามเหลี่ยม A



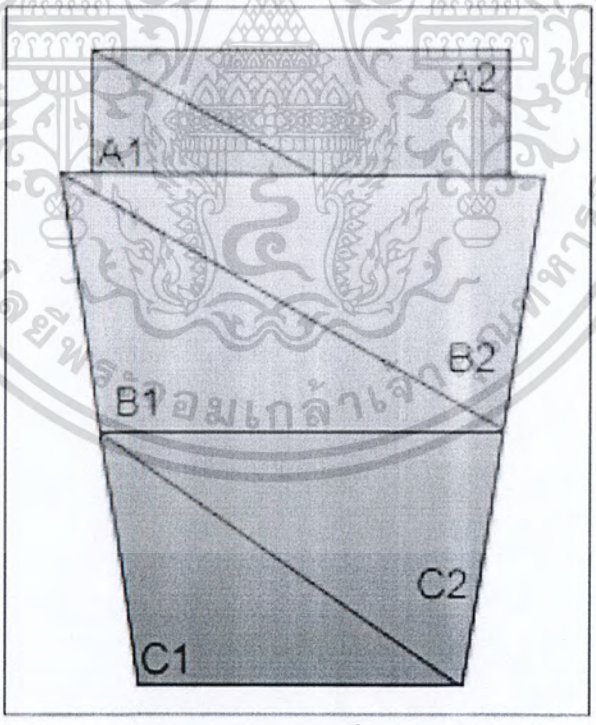
รูป 4.13 ภาพที่คาดหวังไว้และภาพที่ใช้ painter's algorithm

สถานการณ์นี้จะสามารถแก้ไขได้โดยหลากหลายเทคนิค เทคนิคพื้นฐานง่ายๆ แต่มีประสิทธิภาพมากสำหรับสถานการณ์นี้โดยจะมีการแบ่งย่อย polygon B ดังนั้นจะมีสามเหลี่ยมที่อยู่ด้านหน้าของสามเหลี่ยม A และสามเหลี่ยมที่มีด้านหลังของสามเหลี่ยม A นี้เป็นส่วนรายละเอียดเพิ่มเติมและทำให้การจัดลำดับมีความถูกต้องเพิ่มเติม



รูป 4.14 การแบ่งย่อยของภาพสามเหลี่ยม B

ภาพสามเหลี่ยม B ได้ถูกแบ่งออกเป็นสองภาพสามเหลี่ยมคู่ใหม่เป็น B และ C ลำดับการแสดงฉากนี้จะทำให้การวาดภาพสามเหลี่ยม C เป็นอันดับแรกตามด้วยภาพสามเหลี่ยม A และ ภาพสุดท้ายเป็นภาพสามเหลี่ยม B หน้า จะมีผลลัพธ์ดังภาพ



รูป 4.15 การแบ่งย่อยภาพสามเหลี่ยมและจัดอันดับใหม่

แต่ในอเวย์ทรีดีรันอัตโนมัติโดยเรียงลำดับวัตถุในฉาก ทำให้มั่นใจว่าวัตถุ 3 มิติที่แสดงนั้นถูกต้องโดยไม่เกิดปัญหาหารทับซ้อน

4.2.2.3.3 Perspective Projection

เป็นกลไกที่จะแปลงพิกัด 3 มิติของวัตถุ 3 มิติในฉาก ไปเป็นพิกัด 2 มิติ ซึ่งจำเป็นสำหรับการเขียนวัตถุ 2 มิติ ในมุมมอง ในเอย์ทรีดีขั้นตอนนี้จะจัดการโดยคลาสเลนส์ที่พบในแพคเกจ away3d.cameras.lenses เลนส์เช่นเดียวกันนี้จะถูกกำหนดให้เป็นคุณสมบัติเลนส์ของกล้องในการดำเนินการฉายมุมมองในฉาก มีหลายเลนส์ซึ่งแตกต่างกันทำให้สามารถใช้เทคนิคการประมวณผลแตกต่างกัน อย่างไรก็ตามกระบวนการมาตรฐานจะใช้แกน X และ Y ของเส้นฉากและแบ่งมัน โดยแกน Z ที่จะประกอบด้วย X และ Y ของมุมมองเวกเตอร์ (หรือที่เรียกว่าเวกเตอร์หน้าจอ) กระบวนการนี้เป็นสิ่งสำคัญของมุมมองที่ต้องมีคุณสมบัติ เมื่อเพิ่มระยะทางมากขึ้น วัตถุจะมีขนาดเล็กลง

4.2.3 การตั้งค่าพื้นฐานของคลาส

การจัดการกับมุมมองของวัตถุ ฉาก และการใช้งานกล้องในเอย์ทรีดีจะเริ่มต้นด้วยการสร้างคลาสพื้นฐาน ตัวอย่างทั้งหมดนี้จะสร้างต่อจาก Chapter03SampleBase ในแพคเกจ flash3dbook.ch03 ด้วยโค้ดดังตัวอย่าง 4.16

ตัวอย่าง 4.16

```
package flash3dbook.ch03
    {import away3d.cameras.*;
     import away3d.containers.*;
     import away3d.materials.*;
     import away3d.primitives.*;
     import flash.display.*;
     import flash.events.*;
     [SWF(width="800", height="600")]
     public class Chapter03SampleBase extends Sprite
         {protected var _view : View3D;
          protected var _cube1 : Cube;
          protected var _cube2 : Cube;
          public function Chapter03SampleBase()
              {_createView();
               _createScene();
               _createCamera(); }
```

ตัวอย่าง 4.16 (ต่อ)

```
        protected function _createView() : void
        {// Create view and add it to the stage
        _view = new View3D();
        addChild(_view);
        //Relocate center point of view to the center of an 800x600 stage.
        _view.x = 400;
        _view.y = 300;
        //call the view render method on every frame of the Flash movie
        addEventListener(Event.ENTER_FRAME, _onEnterFrame); }
    protected function _createScene() : void
        {// Create a new scene containing a trident and two cubes
        var scene : Scene3D = new Scene3D();
        var trident : Trident = new Trident(200, true);
        _cube1 = new Cube();
        _cube1.x = -100;
        _cube1.material = new WireColorMaterial(0xFFFFFF);
        _cube2 = new Cube();
        _cube2.x = 100;
        _cube2.material = new WireColorMaterial(0x888888);
        scene.addChild(trident);
        scene.addChild(_cube1);
        scene.addChild(_cube2);
        //Assign the new scene to the view
        _view.scene = scene;  }
    protected function _createCamera() : void
        { }
    protected function _onEnterFrame(ev : Event) : void
        { _view.render(); }
    }
}
```

จะมีการสร้างคลาสพื้นฐานด้วย _createView () _createScene () _createCamera () และ _onEnterFrame () สามตัวแรกจะถูกเรียกจากคอนสตรัคเตอร์กับ _onEnterFrame () วิธีการตั้งค่านี้เป็นตัวจัดการสำหรับเหตุการณ์ ENTER_FRAME ส่วน _createView () และ _createScene () จะเริ่มต้นจากโค้ดบางตัวที่กำหนดไว้แล้ว เพื่อที่จะได้รับการตอบสนองบางอย่างดังตัวอย่างก่อนหน้านี้

4.2.4 การสร้างและใช้มุมมอง

ในอเวย์ทรีดีคลาส View3D ตั้งอยู่ในแพคเกจ away3d.containers เพื่อแสดงถึงมุมมองโดยขยายคลาส Sprite และแสดงโดยคลาส instantiating จะเพิ่มในรายชื่อของแฟลชโดย addChild () เช่นเดียวกับการแสดงผลวัตถุใดๆ ตัวอย่างเช่นในไม่กี่บรรทัดแรกของ _createView () ในคลาส Chapter03SampleBase ดังตัวอย่าง 4.17

ตัวอย่าง 4.17

```
// Create view and add it to the stage
_view = new View3D();
addChild(_view);
```

สำหรับฉากและวัตถุที่จะทำให้มองเห็นภาพได้ จะต้องสร้างวัตถุเป็นมุมมองโดยการ render () ใน View3D โดยปกติการ render () ควรจะดำเนินการแค่ครั้งเดียวต่อเฟรม เพื่อให้การปรับปรุงใดๆ ในฉากสามารถทำซ้ำได้อย่างต่อเนื่อง ด้วยเหตุนี้การ render () มักจะอยู่ภายในตัวจัดการเหตุการณ์ ENTER_FRAME เหมือนใน _onEnterFrame ()ดังตัวอย่าง 4.18

ตัวอย่าง 4.18

```
protected function _onEnterFrame(ev : Event) : void
{  _view.render(); }
```

_onEnterFrame () ถูกตั้งค่าให้เป็นตัวจัดการสำหรับเหตุการณ์ ENTER_FRAME ที่ท้ายคลาส _createView ()ดังตัวอย่าง 4.19

ตัวอย่าง 4.19

```
//call the view render method on every frame of the Flash movie
addEventListener(Event.ENTER_FRAME, _onEnterFrame);
```

เช่นเดียวกับการสร้างกล้องอัตโนมัติ และฉากบน instantiation มุมมองยังสร้างจุดศูนย์กลาง และ clipping พารามิเตอร์ และต้องเปลี่ยนแปลงเมื่อวัตถุมีระยะใกล้มากขึ้น โดยดูว่าจะเกิดอะไร เมื่อมีการเปลี่ยนแปลง

4.2.4.1 จุดลับสายตา

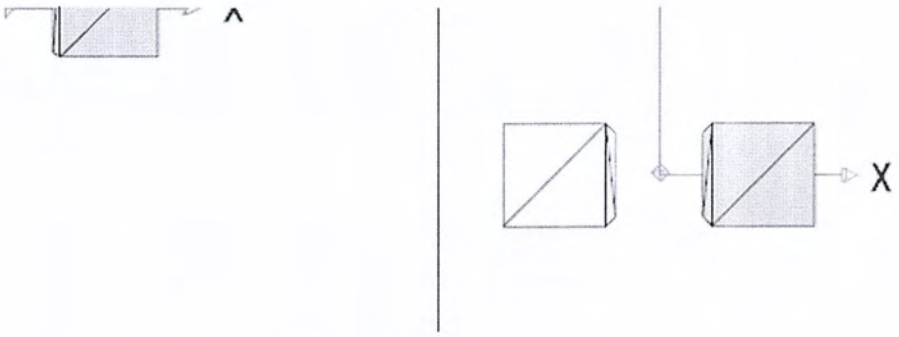
ตำแหน่ง x และ y ของมุมมองบนฉาก จะกำหนดจุดศูนย์กลางสำหรับทุกมุมมองในกราฟิก 3 มิติ จุดศูนย์กลางจะถูกเรียกว่าจุดลับสายตา และเป็นตัวแทนของตำแหน่งที่วัตถุมาบรรจบกันเมื่อไกลออกไป จุดศูนย์กลางยังหมายถึงเวกเตอร์เริ่มต้นสำหรับหน้าจอที่ฉายในฉาก

โดยปกติแล้วจุดศูนย์กลางของมุมมองจะอยู่ที่ (0,0) ที่มุมบนด้านซ้ายของหน้าจอแสดงผล เมื่อต้องการความสมมาตรของวัตถุ 3 มิติ วิธีที่ดีที่สุด คือการย้ายจุดกึ่งกลางของมุมมองที่ศูนย์กลางของวิวพอร์ต(ในกรณีนี้ที่กลางหน้าจอแสดงผล) จะทำโดยการตั้งค่า x และ y เช่นเดียวกับวัตถุอื่นๆ หลังจากการ instantiation ตามที่เห็นใน _createView() ในคลาส Chapter03SampleBase ดังตัวอย่าง 4.20

ตัวอย่าง 4.20

```
//Relocate the center point of the view to the center of an 800x600 stage
_view.x = 400;
_view.y = 300;
```

ในรูปด้านล่างแสดงให้เห็นผลของการไม่ได้ทำขั้นตอนนี้ คือ การคอมไพล์คลาส Chapter03SampleBase กับคำสั่งก่อนหน้านี้จะแสดงผลออกมาตามรูปที่แสดงอยู่ทางด้านซ้าย จุดลับสายตา (จุดเริ่มต้นของฉาก) จะอยู่ในตำแหน่งมุมบนด้านซ้ายของหน้าจอแสดงผล และวิวพอร์ตสามารถแสดงเพียงด้านล่างของลูกบาศก์ หากทำการตั้งค่าแกน x และแกน Y ที่ครึ่งหนึ่งของความกว้างและความสูง และคอมไพล์เพื่อดูผล ดังที่แสดงทางด้านขวา ตอนนี้ทั้งฉากสามารถมองเห็นได้โดยจุดศูนย์กลางจะอยู่ที่ตรงกลางหน้าจอแสดงผล รูปด้านซ้ายจุดศูนย์กลางของมุมมองยังไม่มีเปลี่ยนแปลงในตำแหน่งเริ่มต้น(0,0) ด้านขวาแสดงมุมมองเมื่อมีการตั้งค่าที่ตำแหน่งที่กลางหน้าจอแสดงผล



รูป 4.16 เปรียบเทียบการตั้งค่าจุดศูนย์กลาง

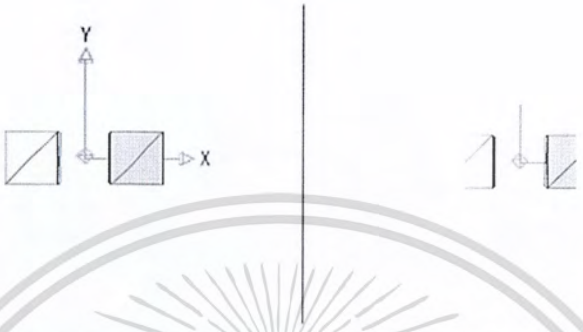
4.2.4.2 การตั้งมุมมองในวิวพอร์ต

ตามที่กล่าวไว้ว่า การเลือกมุมมองบางส่วนให้แสดงผลนั้น จะอยู่ในหน้าต่างที่เป็นวิวพอร์ตกระบวนการของการเปลี่ยนมุมมองวิวพอร์ตเรียกว่าการ clipping และในอเวย์ทรีดีจะควบคุมโดยหนึ่งในคลาสที่มีอยู่ในแพคเกจ away3d.core.clip โดยเช่นเดียวกับฉาก และกล้อง การตั้งค่าเริ่มต้นจะถูกสร้างขึ้นในวัตถุ instantiation โดยขอบเขตของวิวพอร์ตจะมีการใช้ตัวแปร minx maxx minY และ maxY เป็นการกำหนดขนาด สามารถตั้งค่าใหม่เป็นค่าที่กำหนดขอบเขตสูงสุดและต่ำสุดของวิวพอร์ตในทิศทาง x และ y โดยวัดจากจุดลับสายตา (ตำแหน่ง (X, Y) ของวัตถุดูบนหน้าจอแสดงผล) ดังตัวอย่าง 4.21

ตัวอย่าง 4.21

```
package flash3dbook.ch03
    { import away3d.core.clip.*;
      [SWF(width="800", height="600")]
      public class ViewportClipping extends Chapter03SampleBase
        { protected override function _createView() : void
          {super._createView();
           //defining a viewport size of 200 x 200 pixels
           _view.clipping.minX = -100;
           _view.clipping.maxX = 100;
           _view.clipping.minY = -100;
           _view.clipping.maxY = 100; } } }
```

โค้ดนี้ตั้งค่าขอบเขตของการ Clippingไว้ + / - 100 พิกเซล ซึ่งก็คือการกำหนดวิวพอร์ตขนาด 200 x 200 ของคลาส Chapter03SampleBase เมื่อทำการคอมไพล์จะแสดงผลดังรูปในด้านขวา โดยรูปทางซ้ายจะเป็นค่าเริ่มต้นของมุมมองที่แสดงขอบเขตทั้งหมดโดยไม่ได้ทำการ Clipping



รูป 4.17 เปรียบเทียบการแสดงผลเมื่อมีการกำหนดวิวพอร์ต

วิธีการ Clipping จะเกิดจากการตั้งค่าคุณสมบัติ Clipping ที่มีต่อ View3D โดยใช้อินสแตนซ์ RectangleClipping ใหม่ ซึ่งมีคุณสมบัติของขอบเขตการตั้งค่าแล้ว เมื่อต้องการแสดงให้แทนที่ _createView ()ไปแทน ViewportClipping ดังตัวอย่าง 4.22

ตัวอย่าง 4.22

```
protected override function _createView() : void
{       super._createView();
        //defining a new clipping object
        var clipping:RectangleClipping = new RectangleClipping();
        clipping.minX = -100;
        clipping.maxX = 100;
        clipping.minY = -100;
        clipping.maxY = 100;
        //resetting the clipping instance on the view
        _view.clipping = clipping }
```

เมื่อคอมไพล์จะแสดงผลเหมือนในรูปทางขวามือ ดังเช่นที่ผ่านมา

4.2.5 Managing the scene การจัดการฉาก

ตามที่ระบุไว้ก่อนหน้านี้ ฉากเป็นสิ่งที่อเวย์ทรีดีสร้างขึ้นมาเพื่อการแสดงใน เพื่อให้วัตถุแฟลช 3 มิติใช้สำหรับการแสดงผลก็จะต้องมีการเพิ่ม Scene3D ในลักษณะเดียวกับ Sprite ที่ต้องมีการบันทึกอยู่ในด็อคคิดเม้นเมื่อทำการฉายภาพ นี่คือสิ่งที่ต้องทำเพื่อให้ได้ลูกบาศก์ และทรงกลมในการ _createScene ของคลาส Chapter03SampleBase วัตถุใน Scene3D มีลำดับชั้นของวัตถุ 3 มิติที่ซ้อนกัน เช่นเดียวกับการแสดงภาพในแฟลชซึ่งรู้จักในชื่อของฉาก

4.2.5.1 การเพิ่มและลบวัตถุ 3D

ส่วนใหญ่ของวัตถุกับการแสดงผลในอเวย์ทรีดีรับช่วงจากพื้นฐานของคลาส Object3D วัตถุประเภทนี้ถูกกำหนดเป็นโหนดในฉาก และสามารถเพิ่มโดยการใช้ addChild ของฉาก สังเกตุ _createScene() ในคลาส Chapter03SampleBase สามารเห็นว่าวัตถุ 3 มิติที่แสดงผลนั้นมีการเพิ่มได้เพียงครั้งละหนึ่งในลักษณะเช่นนี้ ดังตัวอย่าง 4.23

ตัวอย่าง 4.23

```
scene.addChild(trident);
scene.addChild(_cube1);
scene.addChild(_cube2);
```

วิธีการ addChild (และวิธีการอื่นๆ ที่เกี่ยวข้องกับการจัดการของการสืบทอดวัตถุ) ที่สืบทอดจากคลาส ObjectContainer3D จากการขยายคลาส Scene3D ซึ่งเป็นที่เก็บคลาสสำหรับฉากที่สามารถใช้ในการสร้างลำดับชั้นที่ซ้อนกันสำหรับวัตถุที่มีอยู่ภายในฉาก คลาส ObjectContainer3D ขยายคลาสพื้นฐาน Object3D ซึ่งมีความคล้ายกันกับแฟลชดังรูปที่เปรียบเทียบระหว่างลำดับคลาสในแฟลช(ซ้าย) และฉากลำดับคลาสในอเวย์ทรีดี(ขวา)

รูป 4.18 เปรียบเทียบลำดับคลาสระหว่างอเวย์ทรีดีและ แฟลช

สร้างตัวอย่างใหม่ที่ขยายคลาส Chapter03SampleBase เพื่อตรวจสอบคุณสมบัติขอ Scene3D และวัตถุ ObjectContainer3D ดังตัวอย่าง 4.24

ตัวอย่าง 4.24

```
package flash3dbook.ch03
    {
    import away3d.primitives.Cube;
    import away3d.containers.ObjectContainer3D;
    import away3d.core.base.Object3D; [SWF(width="800",          height="600")]
    public class SceneProperties extends Chapter03SampleBase
        {
        protected override function _createScene() : void
            { super._createScene();
            _view.scene.removeChild(_cube2); // Remove from scene
        } } }
```

ในการลบวัตถุที่มีอยู่ในฉากออกจากฉาก สามารถใช้ removeChild (รับการถ่ายทอดมาจาก ObjectContainer3D) วัตถุที่สืบทอดจะถูกลบออกจาก parent ของตัววัตถุเอง โดยการสั่งฟังก์ชั่นนี้ในคลาสของ parent โค้ดในก่อนหน้านี้ ใช้ removeChild () เพื่อลบลูกบาศก์ที่สอง (สีเทา) จากฉาก ก่อนที่จะแสดงในมุมมอง การคอมไพล์ตัวอย่างนี้จะแสดงเพียงก้อนแรก (สีขาว) ว่าวัตถุถูกลบจากหน้าจอที่แสดงผล

4.2.5.2 การเข้าถึงวัตถุ 3 มิติในฉาก

วิธีหนึ่งในการเข้าถึงคลาสลูกของ ObjectContainer3D คือการใช้คุณสมบัติของตัวคลาสลูกเอง คือจะส่งกลับอาร์เรย์ที่มีวัตถุ 3 มิติอยู่คืนทั้งหมด ซึ่งคลาสลูกนั้นๆสืบทอดมา และสามารถใช้ในการกำหนดข้อมูลที่เป็นประโยชน์บางอย่างเกี่ยวกับวัตถุที่มีอยู่ในคลาสลูก ตัวอย่างเช่น คุณสมบัติความยาวของอาเรย์ สามารถนำมาใช้เพื่อดึงจำนวนรวมของวัตถุที่คลาสลูกสืบทอดมา การเพิ่มโค้ดใน _createScene () ซึ่งอยู่ใน SceneProperties เขียนโค้ด และทำการคอมไพล์จะมีการส่งกลับค่า 3 จำนวนไปที่หน้าต่างแสดงผล ของคลาสลูกที่สืบทอดมา เพราะวัตถุทั้งสามเป็นคลาสลูกของฉาก ตัวอย่าง 4.25

ตัวอย่าง 4.25

```
protected override function _createScene() : void
{   super._createScene();
    trace(_view.scene.children.length); }
```

การเรียกคืนวัตถุ 3 มิติในฉาก ถ้าเป็นคำสั่งที่สืบทอดอยู่ในคลาส 3 มิติ ทำได้โดยการเข้าถึงตำแน่งที่ถูกต้องของคลาสลูก โดยการเพิ่มโค้ดใน _createScene () ซึ่งอยู่ใน SceneProperties ตัวอย่างเช่นโค้ดนี้ เมื่อทำการคอมไพล์ จะมีการส่งกลับค่าบอกว่าจริงยังหน้าต่างแสดงผล เนื่องจากคลาสลูกตัวแรกที่ส่งไปยังฉากเป็นลูกบาศก์ ดังตัวอย่าง 4.26

ตัวอย่าง 4.26

```
protected override function _createScene() : void
{   super._createScene();
    trace(_view.scene.children[0] is Cube); // returns true  }
```

วัตถุ 3 มิติในฉากยังสามารถเรียกโดยใช้ชื่อซึ่งเป็นสตริง ID ที่ไม่ซ้ำกัน โดยจะตั้งค่าก่อนที่จะเพิ่มวัตถุลงในฉาก เมื่อเพิ่มข้อมูลแล้ว วัตถุ 3 มิติจะเรียกใช้ getChildByName() ในฉากโดยการเพิ่มโค้ดใน _createScene () ซึ่งอยู่ใน SceneProperties ทำการคอมไพล์ จะมีการส่งกลับค่าบองว่าเป็นจริง เพราะ theFirstCube เป็นชื่อของ วัตถุ _cube1 ในฉาก ดังตัวอย่าง 4.27

ตัวอย่าง 4.27

```
protected override function _createScene() : void
{   super._createScene();
     _cube1.name = "theFirstCube";
    trace(_view.scene.getChildByName("theFirstCube") is Cube); // returns true }
```

ข้อดีอย่างหนึ่งของการใช้ getChildByName () คือเป็นวิธี recursive ธรรมชาติ วิธีนี้สามารถนำมาใช้ เพื่อส่งกลับวัตถุ 3 มิติจากคลาสแม่ในฉาก แม้ว่าจะถูกฝังอยู่ใน subcontainers

ในการเพิ่มวัตถุ 3 มิติไม่ใช่เรื่องง่าย ที่จะเพิ่มไปยังตำแหน่งเฉพาะของอาเรย์ลูก เพราะคำสั่งเฉพาะของวัตถุมีความสำคัญไม่มากในฉาก ในอเวย์ทรีดีขั้นตอนของ Z-sorting จะกำหนดลำดับของวัตถุที่ส่งมายังหน้าจอโดยอัตโนมัติ

ส่วนในแฟลชนั้น ในพื้นที่การแสดงภาพ ไม่มีการทำ Z-sorting ดังนั้นคำสั่งในการเพิ่มวัตถุจะต้องมีการเรียงลำดับเข้ามา การอธิบายนี้เป็นการอธิบายว่าทำไมคำสั่ง เช่น getChildIndex () และ setChildIndex () จะถูกตัดออกจากคลาส ObjectContainer3D และคลาส Scene3D

4.2.5.3 การทำงานกับวัตถุ 3 มิติที่ซ้อนกัน

ดังเช่นที่เห็น ฉากในการรองรับวัตถุ ObjectContainer3D สืบทอดจาก Object3D และสามารถเพิ่มเป็นคลาสลูกของคลาสอื่นจะช่วยให้วัตถุที่ซ้อนกันเป็นไปตามที่ต้องการ ตัวอย่างเช่นเปลี่ยนคำสั่งใน _createScene () ซึ่งอยู่ใน SceneProperties โค้ดนี้จะส่งค่ากลับเป็นจริงทั้งหมด ดังตัวอย่าง 4.28

ตัวอย่าง 4.28

```
protected override function _createScene() : void
{   super._createScene();
    _cube1.name = "theFirstCube";
    var myGrandParent : ObjectContainer3D = new ObjectContainer3D();
    var myParent : ObjectContainer3D = new ObjectContainer3D();
    var mySelf : Object3D = new Object3D;
    myParent.addChild(mySelf);
    myGrandParent.addChild(myParent);
    _view.scene.addChild(myGrandParent);
    trace(mySelf.parent == myParent); // prints true
    trace(mySelf.parent.parent == myGrandParent); // prints true
    trace(myGrandParent.children[0].children[0] == mySelf); // prints true }
```

ในการสืบทอดนั้นการทอดถ่ายจะยอมรับแค่สองลำดับขั้น โดยคุณสมบัติของคลาสแม่จะมีอยู่ใน Object3D และชี้ไปยังอินสแตนซ์ ObjectContainer3D ที่คลาสนั้นอยู่ วัตถุสามารถมีคลาสแม่ได้เพียงคลาสเดียว ถ้าใช้คำสั่ง addChild () จะทำในวัตถุที่มีอยู่แล้วไปยังที่อื่นใน

ฉาก ซึ่งจะถูกลบออกจากที่นั้น ก่อนที่จะถูกเพิ่มเข้าไปในคลาสใหม่ ในบรรทัดสุดท้ายของโค้ดก่อนหน้านี้ ทำการเข้าถึงโดยตรงไปยังอาเรย์ลูก เพื่อส่งค่ากลับไปยังคลาสแรก ซึ่งได้รับการยืนยัน ว่าตัวเองที่เป็นหลานของ myGrandParent

การลบวัตถุ 3 มิติจากฉากคือ removeChild () วิธีการนี้จะต้องมีการดำเนินการในทันทีกับคลาสแม่ของวัตถุ เพิ่มโค้ดนี้ไปยัง _createScene () วิธีการข้างต้นจะทำงานเพราะ myself เป็นคลาสลูกของ myParent ดังตัวอย่าง 4.29

ตัวอย่าง 4.29

```
myParent.removeChild(mySelf); // Works, myParent is the parent of mySelf
```

แต่โค้ดนี้จะล้มเหลว เพราะ mySelf ไม่ใช่ลูก myGrandParent ดังตัวอย่าง 4.30

ตัวอย่าง 4.30

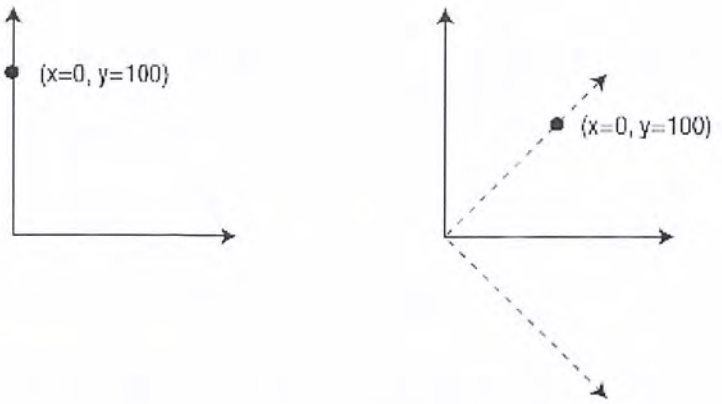
```
myGrandParent.removeChild(mySelf); // Fails
```

4.2.5.4 การย้าย การหมุน และการปรับขนาดใน 3D

ในฉากที่แสดงนั้น พิกัดของวัตถุ 3 มิติ โดยทั่วไปจะไม่ได้ล็อคพิกัด แต่ได้รับอิทธิพลจากวัตถุของคลาสแม่ในลำดับชั้น แนวคิดนี้สามารถทำให้เกิดความสับสนในตอนแรก แต่ในที่สุดจะเห็นว่าวิธีนี้สามารถนำไปใช้ประโยชน์ เมื่อมีการดำเนินงานที่ซับซ้อนมากขึ้น

แนวคิดของระบบพิกัดไม่มีอะไรใหม่ และในแฟลชเป็นสิ่งที่ใช้ในการทำงาน 2 มิติอยู่บ่อยครั้ง ถ้าให้สร้าง MovieClip ในแฟลชหมุน 45 องศา และสร้าง MovieClip อีกอันภายใน MovieClip ก่อนหน้านี้ และหมุน 45 องศาเช่นกัน จะสังเกตได้ว่าคลิปด้านในจะปรากฏการหมุน 90 องศา และจะส่งกลับค่า 45 ซึ่งไม่ถือเป็นความผิดพลาด สิ่งที่กำลังเห็นนี้เป็นระดับที่แตกต่างกันของการหมุนในระบบพิกัดที่มีการกำหนดพิกัดการแปลงของคลาสลูกในลักษณะสะสม ตามที่กล่าวมานี้ คือการเรียนรู้เป็นระบบพิกัดลำดับชั้น และเป็นซอฟแวร์ที่ใช้ในการออกแบบกราฟิกที่มีการใช้อยู่บ่อยครั้ง

การเพิ่มมิติทำให้ยากที่จะปฏิบัติตาม เมื่อเกิดการถ่ายทอดไปยังคลาสลูก ดังนั้นในตอนนี้จะพูดถึงในแง่ของ 2 มิติ โดยลองนึกภาพพิกัดจุดบนแกน Y ใน 2 มิติ เนื่องจากอยู่บนแกน Y ทำให้รู้ว่าค่าพิกัดบนแกน X เป็น 0 ดังเช่นภาพที่อยู่ทางด้านซ้ายในรูป



รูป 4.19 การเปลี่ยนจุดในระบบพิกัดจะส่งผลต่อทุกแกน

ถ้าหมุนระบบพิกัดของจุดไป 45 องศา ภายในฉากที่วัตถุแสดงจะเปลี่ยนไป รูปจะเหมือนทางด้านขวาตำแหน่งในทุกแกนจะไม่ตรงกับตำแหน่งในคลาสลูก แม้ว่าเวกเตอร์ตำแหน่งของคลาสลูกจะไม่ได้เปลี่ยนแปลง

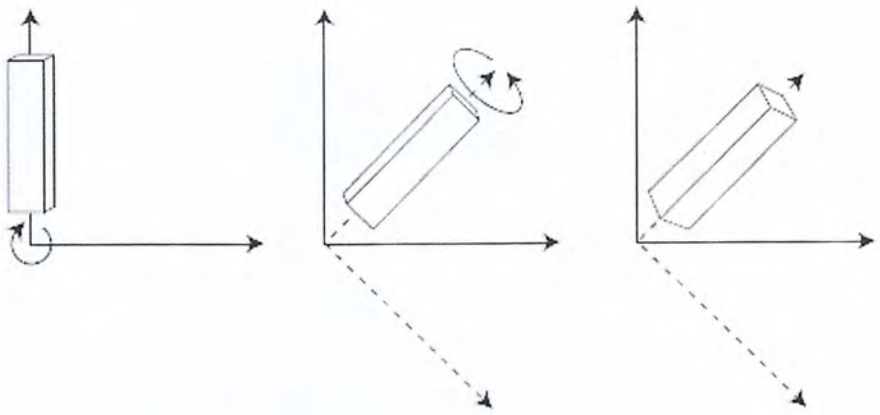
การทำความเข้าใจแนวคิดนี้ใน 2 มิติ สามารถใช้ขั้นตอนเป็น 3 มิติโดยการดูวิธีการหมุนแกนของ ObjectContainer3D ซึ่งส่งผลกระทบต่อระบบการประสานงานของคลาสย่อย สำหรับใน 2 มิติจะมีตัวเลือกการหมุนในระนาบของพื้นผิว (รอบแกน Z ทางทฤษฎี) ใน 3 มิติมีตัวเลือกของการหมุนรอบแกน คือ X Y Z หรืออาจจะหมุนทั้ง 3 แกนพร้อมกัน

ในรูปด้านล่างนี้ ยึดการคิดเพียงสองแกน เพื่อความชัดเจน (แกน X และแกน Y) แต่เวลานี้ จะทำการหมุนสองขั้นตอนรอบแกน Z และแกน Y ตามลำดับ ภาพทางด้านซ้ายเป็นตัวแทนของจอแสดงผลที่เริ่มต้นกับบล็อก 3 มิติแนวแกน Y และมีศูนย์กลางอยู่ที่แกน X และแกน Z

ในขั้นตอนแรก ใช้การหมุนเช่นเดียวกับที่ใช้ในรูปที่ 4.19 โดยหมุน 45 องศารอบแกน Z (ซึ่งสามารถจินตนาการเป็นเส้นตั้งฉากชี้ไปด้านหน้า) ของจอแสดงผล 3D เมื่อเสร็จสิ้นการหมุนแกน Y จะชี้ขึ้นและเยื้องไปทางขวา การหมุนต่อไปนี้ยังคงสัมพันธ์กับคลาสแม่ของระบบพิกัด และสิ้นสุดโดยชี้ขึ้นไปยังทิศทางที่ปรากฏในภาพกลางของรูปที่ 4.20

ขั้นตอนที่สองเป็นขั้นตอนหนึ่งที่มีความสำคัญต่อความเข้าใจธรรมชาติของระบบพิกัดลำดับชั้น จะทำการหมุนบล็อกไป 45 องศารอบแกนของตัวเองในคลาสย่อยของแกน Y เพราะครั้งแรกหมุนระบบพิกัด 3 มิติของจอแสดงผล การหมุนจะถูกดำเนินการเปลี่ยนภาพแสดงของแกน Y ลูกศรหมุนเวียนในภาพกลางรูปที่ 4.20 แสดงให้เห็นถึงการหมุน

จากภาพจะแสดงให้เห็นถึง การหมุนสองครั้งติดต่อกันของบล็อกสี่เหลี่ยมในระบบพิกัด คลาสย่อยของแกน Y จะถูกหมุนรอบแกน Z ในครั้งแรก ซึ่งการทำงานนี้มีผลต่อการหมุนรอบแกน Y ในขั้นตอนที่ 2



รูป 4.20 การหมุนของแกนที่ส่งผลกระทบต่อกัน

ผลสุดท้ายหลังจากมีการหมุนบล็อคไป 45 องศารอบแกนของตัวเองในคลาสย่อยของแกน Y จะปรากฏตามภาพด้านขวาของรูปที่ 4.20

เมื่อหมุนรอบวัตถุ 3 มิติมากกว่าหนึ่งแกน คำสั่งดำเนินการหมุนเป็นสิ่งสำคัญ การหมุนก่อนอาจมีผลต่อทิศทางของแกนที่ใช้ในการหมุนภายหลัง ปัญหานี้เป็นปัญหาที่พบบ่อยใน 3 มิติที่เรียกว่า gimbal lock

4.2.5.5 หลักของการเพิ่มคลาสสืบทอด

จากเนื้อหาในก่อนหน้านี้ คงทำให้คุ้นเคยกับการทำงาน 3 มิติที่มีการซ้อนกันของจอแสดงผล ในขั้นตอนต่อไปจะเป็นการนำไปใช้ประโยชน์ในฉากของอเวย์ทรีดีหากผู้ใช้ไม่ได้หลักคณิตศาสตร์ของ 3 มิติจะพบว่าการหมุนวัตถุ 3 มิติ รอบจุดจุดหนึ่งในพื้นที่ (เช่นการจำลองของโลกหมุนรอบดวงอาทิตย์หรือลูกตุ้มควงรอบหมุนของมัน) ไม่ได้งานที่ง่าย แต่ก็สามารถทำได้หากใช้ประโยชน์ของระบบพิกัดลำดับชั้นของฉาก

ในอเวย์ทรีดีคุณสมบัติ rotation rotationY และ rotationZ ใช้การหมุนวัตถุ 3 มิติรอบแกน X, Y และ Z ตามลำดับ สำหรับวัตถุรูปทรงเรขาคณิตที่สร้างขึ้นภายใน (เช่นลูกบาศก์ ทรงกลม) ที่มาของพื้นฐานการประสานงานซึ่งการหมุนจะดำเนินงานที่จุดกลางของเรขาคณิต ตัวอย่างเช่นเมื่อหมุนทรงกลมรอบแกนใด ๆ การหมุนจะดำเนินการรอบจุดศูนย์กลางของทรงกลม การสร้างวัตถุ 3 มิติมีศูนย์กลางอยู่ที่จุดเริ่มต้นในพื้นที่ยังคงเป็นวิธีการทั่วไปในการสร้างแบบจำลอง 3 มิติ ส่งผลให้การนำเข้าไฟล์จำนวนมากมีรูปแบบที่คล้ายๆกัน

อย่างไรก็ตาม ศูนย์กลางของการหมุนไม่ได้อยู่ที่จุดกึ่งกลางของเรขาคณิตเสมอไป ในกรณีของลูกตุ้ม การหมุนต้องมีการทำงานที่จุดปลายด้านหนึ่งของวัตถุ สร้างตัวอย่างในคลาส Chapter03SampleBase ดังตัวอย่าง 4.31

ตัวอย่าง 4.31

```
package flash3dbook.ch03
    {import away3d.containers.*;
     import away3d.primitives.*;
     import flash.events.Event;
     import flash.utils.getTimer;
     [SWF(width="800", height="600")]
     public class PendulumContainer extends Chapter03SampleBase
        {private var _pendulum:ObjectContainer3D;
         protected override function _createScene() : void
            {
            // Create a new scene containing a pendulum
             var scene : Scene3D = new Scene3D();
             var trident : Trident = new Trident(100, true);
             var sphere : Sphere = new Sphere();
             sphere.radius = 20;
             //create a new pendulum container
             _pendulum = new ObjectContainer3D();
            }
        }
    }
```

ในโค้ดก่อนหน้านี้จะแทนที่ _createScene () และสร้างวัตถุใหม่ในฉาก ตรีศูลและทรงกลมพื้นฐานที่มีรัศมีเท่ากับ 20 นอกจากนี้จะสร้างวัตถุ 3 มิติที่เก็บในตัวแปร global เรียกว่า _pendulum ซึ่งสามารถเข้าถึงได้จากที่ใดก็ได้ในคลาส โดยกำหนดตัวอย่างใหม่ของ ObjectContainer3D นี้ จะทำหน้าที่เป็นฉากซึ่งรองรับลูกตุ้ม

ถัดไปจะกำหนดตำแหน่งของทรงกลมโดยให้ y คือ -100 และเพิ่มเป็นลูกของ _pendulum โดยการเพิ่มโค้ดต่อไปนี้ต่อท้ายใน _createScene ซึ่งจะเป็นการสร้างรูปทรงกลมภายในลูกตุ้ม ด้วยพิกัด local -100 หน่วยอยู่ในแกน Y ดังตัวอย่าง 4.32

ตัวอย่าง 4.32

```
//offset the local position of the sphere to (0, -100, 0)
sphere.y = -100;
//add the sphere to the pendulum container
_pendulum.addChild(sphere);
```

กำหนดตำแหน่งอยู่สำหรับลูกตุ้มให้อยู่ในประเภทของ local เพิ่มโค้ดดังตัวอย่าง 4.33 หลังจาก _createScene () เป็นการเพิ่มวัตถุตรีศูลที่สร้างขึ้นใหม่เป็นลูกของ _pendulum

ตัวอย่าง 4.33

```
//add the trident to the pendulum container
_pendulum.addChild(trident);
```

สุดท้าย เพิ่มโค้ดต่อไปนี้ใน _createScene () เพื่อเพิ่มลูกตุ้มให้เป็นลูกของฉาก และกำหนดฉากใหม่ในมุมมองเพื่อให้แสดงผล ดังตัวอย่าง 4.34

ตัวอย่าง 4.34

```
scene.addChild(_pendulum);        //add the pendulum container to the scene
_view.scene = scene;              //Assign the new scene to the view
```

การหมุนทรงกลมที่จุดนี้ก็จะทำให้วัตถุหมุนรอบแกนของตัวเอง แต่ถ้าหมุนที่ลูกตุ้ม ทรงกลมจะแกว่งในส่วนโค้งคล้ายกับการเคลื่อนไหวที่เห็นในรูปที่ 4.20 ที่ตำแหน่ง local วัตถุจะถูกเปลี่ยนแปลงโดยการหมุนของคลาสแม่

การเคลื่อนไหวของลูกตุ้ม สามารถใช้วิธีการ getTimer ในทุกเฟรมของแฟลช ค่าในแนวเส้นตรงที่เพิ่มของการแกว่ง สำหรับการเคลื่อนไหวจริง ฟังก์ชั่น sine สามารถใช้ค่า getTimer () โดยการเพิ่มโค้ดต่อไปนี้ที่ท้ายสุดของคลาส PendulumContainer เพื่อแทนที่ _onEnterFrame () ดังตัวอย่าง 4.35

ตัวอย่าง 4.35

```
protected override function _onEnterFrame(ev : Event) : void
        { _pendulum.rotationZ = 45 * Math.sin(getTimer() / 500);
          _view.render(); }
```

เมื่อคอมไพล์โค้ดทรงกลมจะเคลื่อนไหวโดยการแกว่ง และลูกตุ้มจะแกว่งไปมาระหว่าง 45 และ -45 องศารอบๆจุดกำเนิด สำหรับการเปรียบเทียบ ลองเปลี่ยน pendulum.rotationZ ไปเป็น pendulum.children [0]. rotationZ (rotationZ ของทรงกลม) และคอมไพล์จะเห็นความแตกต่างระหว่างการหมุนลูกตุ้มและหมุนทรงกลมพื้นฐานโดยตรง

การเคลื่อนไหวของลูกตุ้มเหมือนที่เห็นในตัวอย่าง PendulumContainer จะได้สำเร็จโดยใช้ตรีโกณมิติในพื้นที่ 3 มิติที่ซ้อนกันของคลาส แต่วิธีการนี้เป็นวิธีที่ง่าย และสะดวกขึ้น โดยเฉพาะอย่างยิ่งถ้าจะเริ่มสร้างการหมุนที่ซับซ้อนขึ้น

4.2.6 การสร้างและการใช้กล้อง

หลักการของกล้องตามทฤษฎีใน 3 มิติ แล้ว กล้องเป็นแค่จุดในพื้นที่ของ 3 มิติ ซึ่งแสดงผลเป็นจุดในมุมมองของพื้นที่ทั้งหมด การเปลี่ยนตำแหน่งของกล้อง เป็นการเปลี่ยนมุมมองในโลกของ 3d แต่ในความเป็นจริงกล้องไม่ได้เปลี่ยนตำแหน่ง มันเป็นการแทนที่วัตถุทุกชิ้นโดยอ้างอิงถึงตำแหน่งของ camera เท่านั้น

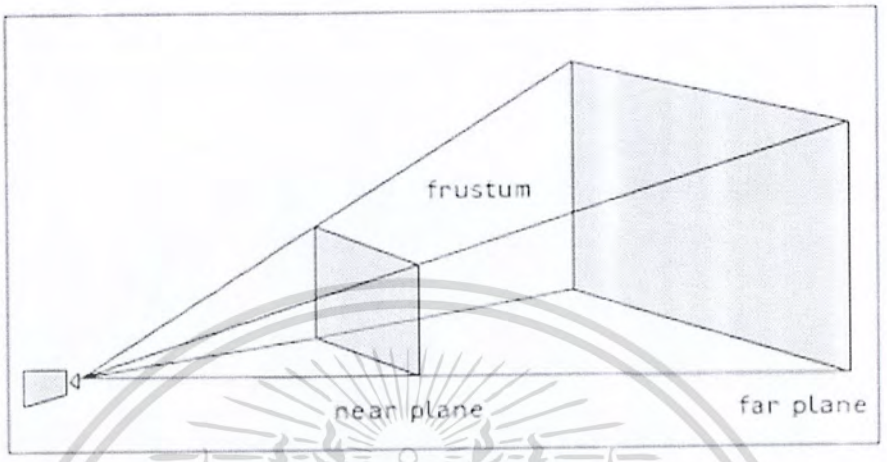
ในอเวย์ทรีดีมีกล้องอยู่หลายประเภท เป็นตัวแทนของคลาสที่พบในแพคเกจ away3d.cameras ในขณะที่งานหลักของกล้องคือการแสดงตำแหน่งวัตถุในพื้นที่ที่เกี่ยวข้องกับฉาก กล้องยังกำหนดการซูม และเน้นคุณสมบัติที่มีผลต่อการประมาณการของฉากต่อมุมมอง ในที่นี้จะบอกรายละเอียดเกี่ยวกับประเภทของกล้องแต่ละตัวในอเวย์ทรีดีวิธีการที่แตกต่างกันของการเคลื่อนไหวและกล้องที่ใช้ขึ้นอยู่กับชนิดของการชมที่ผู้ชมต้องการ

กล้องถ่ายภาพพื้นฐานในอเวย์ทรีดีคือคลาส Camera3D กรณีของการที่จะถูกสร้างขึ้นโดยค่าเริ่มต้นของคุณสมบัติกล้องของวัตถุ View3D กล้องนี้มีมาตรฐาน โดยหันหน้าฉายมุมมองของฉากโดยการจัดตำแหน่งตัวเอง -1,000 หน่วยตามแกน Z และฉายภาพจุดกำเนิดของฉาก

4.2.6.1 พื้นฐานการตั้งค่ากล้อง

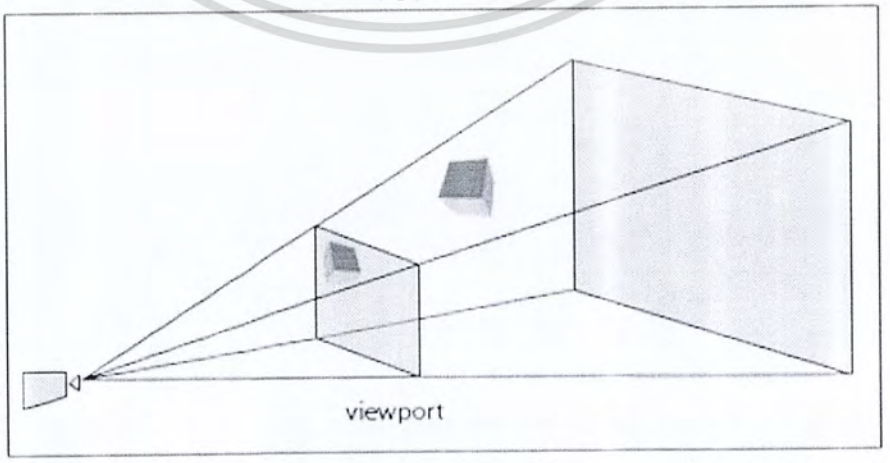
จากทฤษฎีที่กล่าวมาข้างต้นนั้น จะสามารถนำมาตั้งค่าพื้นฐานที่ใช้เฉพาะกับกล้องแบบสรุปโดยมีวิธีการตั้งค่าที่เกี่ยวข้องกัน ดังนี้

ภาพประกอบต่อไปนี้แสดงสิ่งที่กล้องเห็น รูปร่างปิรามิดที่มี far plane near plane และพื้นที่พีระมิดที่อยู่ระหว่างกลางจะเป็น viewing volume หรือ frustum การมี far plane และ near plane เพื่อกำหนดค่าให้ frustum



รูป 4.21 สิ่งที่กล้องในสามมิติเห็น

ความสัมพันธ์เกี่ยวข้องระหว่างfrustum กับวิวพอร์ตที่ทุกๆอย่างภายใน frustum จะถูกวาดลงวิวพอร์ตจะต่างกับการใช้โปรแกรม 3 มิติ ที่มีการเซ็ตวิวพอร์ตไว้ที่แตกต่างกันกันไป ใน Away3D การเซ็ตวิวพอร์ตจะเกิดขึ้นที่ near plane แม้ว่าวิวพอร์ตเป็นsprite สองมิติ และ near plane ระนาบตั้งฉากกับกล้องที่จินตนาการขึ้นทั้งสองมีความสัมพันธ์กันจริง สมมุติให้จินตนาการเพิ่ม cube(ลูกบาศก์) ลงในฉาก ในภาพประกอบข้างล่างนี้สามารถเห็นลูกบาศก์ภายใน frustum ซึ่งเป็นข้อมูล 3 มิติที่ถูกเพิ่ม material ให้แล้ว Away3D จะทำการฉายจุดต่างลงในวิวพอร์ต 2 มิติ และทำการเรนเดอร์ภาพเพื่อให้ผู้ใช้เห็นบนหน้าจอ



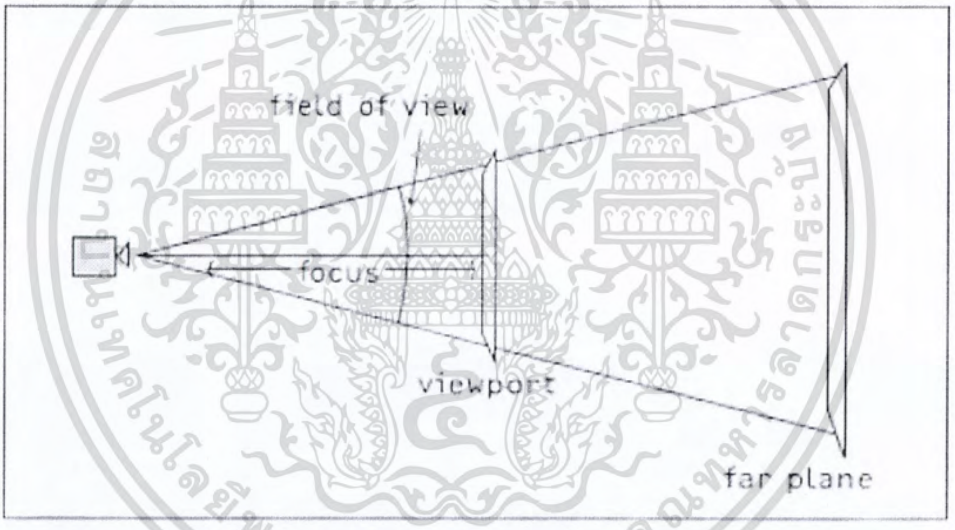
รูป 4.22 การฉายจุดลงในวิวพอร์ตและ การเรนเดอร์(render) ภาพเพื่อให้ผู้ใช้เห็นบนหน้าจอ

frustum เป็นพื้นที่ 3 มิติ ที่ปรากฏบนหน้าจอ ดังนั้นวัตถุที่อยู่ภายนอก frustum จะไม่ปรากฏบนหน้าจอ จะเป็นการคำนวณที่สูญเปล่า ถ้าจะให้ Papervision3D คำนวณวัตถุที่อยู่ภายนอก frustum กระบวนการที่เกิดขึ้นจะเป็นวัตถุที่ถูกระบุอยู่ภายใน frustum และไม่แสดงสิ่งที่อยู่ภายนอกเรียกว่า culling(การคัด) และมีบทบาทสำคัญในการเพิ่มประสิทธิภาพการทำงานของภาพสามมิติ สามารถช่วยประหยัดการประมวลผลได้มาก ถ้า Papervision3D ไม่ประมวลผลจุดทั้งหมดที่ไม่สามารถมองเห็น

4.2.6.1.1 โฟกัส และ field of view

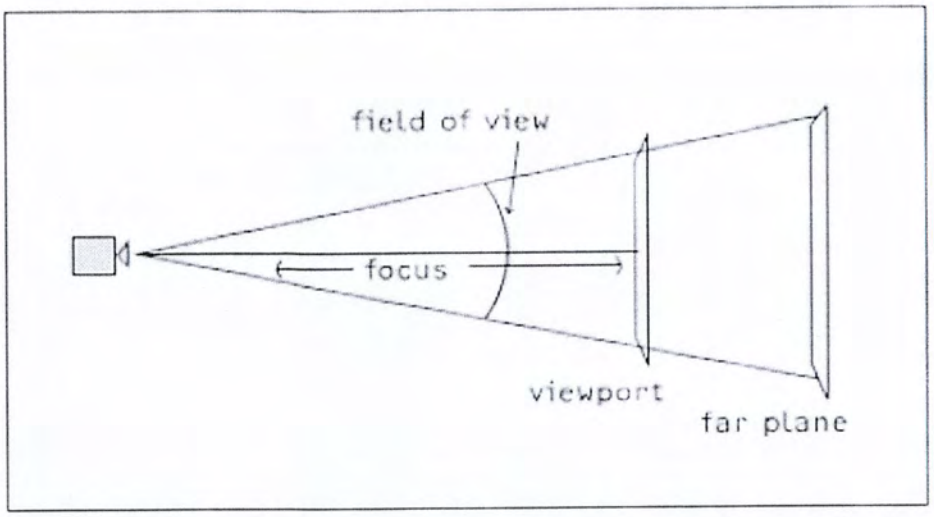
โฟกัสเป็นค่าจำนวนเต็มบวกที่แสดงถึงระยะห่างระหว่างกล้องถ่ายรูปและ near plane (หรือที่เป็นวิวพอร์ต) เป็นระยะทางที่สั้นที่สุดระหว่างผู้สังเกตการณ์กับวัตถุที่มองเห็น

Field of view เป็นมุมกล้องในแนวตั้งของ view



รูป 4.23 ความสัมพันธ์ระหว่าง focus field of view และความสูงของวิวพอร์ต

การเพิ่มโฟกัสขึ้นจะเป็นการเพิ่มระยะห่างระหว่างกล้องกับ near plane ซึ่งส่งผลให้ระยะที่วัตถุจะสามารถปรากฏได้น้อยลง โดยไม่คำนึงถึงระยะห่างระหว่างวัตถุเหล่านั้น นอกจากนี้ field of view จะมีขนาดเล็กลงด้วยดังภาพถัดไป



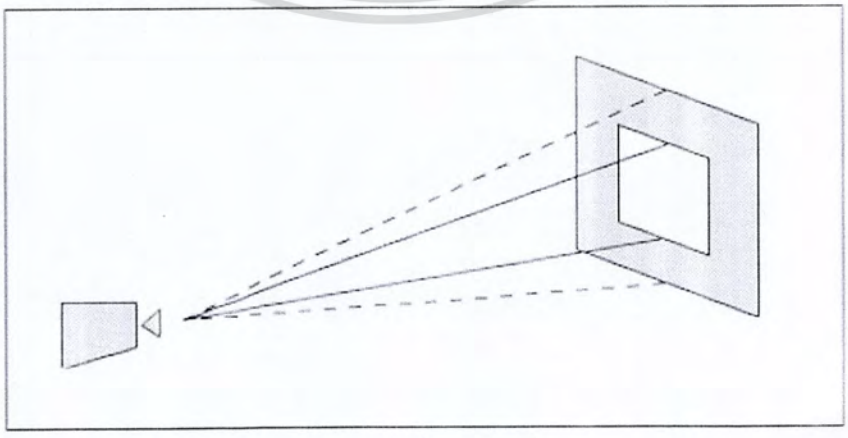
รูป 4.24 ความสัมพันธ์ระหว่าง focus field of view และความสูงของวิวพอร์ตเมื่อเพิ่มโฟกัส

การลดความยาวโฟกัสจะเป็นการเพิ่มขนาดของ field of view ให้มากขึ้น เหมือนการใช้เลนส์ wide จะมีผลทำให้ภาพในมุมมองของวัตถุบิดเบือนไป วัตถุใกล้จะปรากฏขนาดใหญ่และวัตถุไกลปรากฏมีขนาดเล็ก การตั้งโฟกัสต่ำๆ จะเกิดการบิดเบือนมุมมองของภาพแบบ perspective distortion

4.2.6.1.2 Zoom

เมื่อทำการขยายภาพหรือวิดีโอ ภาพฉากทั้งหมดจะขยายขึ้น โดยไม่ต้องย้ายกล้องเข้าไปในฉาก ระยะทางระหว่างวัตถุใกล้ขึ้นเหมือนภาพวัตถุที่ดูเหมือนจะถูกบีบอัดลง เป็นเช่นเดียวกับที่ซูม เพิ่มขยายค่าของกล้องใน Papervision3D

ภาพประกอบต่อไปนี้ การซูมด้วยกล้องเป็นการขยายวัตถุที่จะเห็นในพื้นที่แทนด้วยสี่เหลี่ยมสีขาวเล็กๆ จะเห็นว่าหากให้ความสำคัญเดียวกัน จะต้องให้ field of view เล็กเพื่อทำให้วัตถุที่เห็นปรากฏภาพใหญ่



รูป 4.25 การซูม (zoom) เพื่อขยายหรือลดขนาดวัตถุ

Zoom focus และ field of view มีค่าที่สัมพันธ์ซึ่งกันและกัน หากค่าใดค่าหนึ่งเปลี่ยน จะส่งผลทำให้ค่าอื่นเปลี่ยนไป สามารถสรุปได้ดังนี้

-ถ้าลดค่า zoom หรือโฟกัสค่า field of view จะเพิ่มขึ้น

-ถ้าเพิ่มค่า zoom หรือโฟกัสค่า field of view จะลดลง

-การเพิ่ม หรือการลดค่า field of view จะเป็นการลดหรือเพิ่มค่าโฟกัสตามลำดับ แต่จะไม่มีการเปลี่ยนแปลงของค่า zoom

4.2.6.2 ประเภทของกล้อง(camera)

4.2.6.2.1 ทาร์เก็ตคาเมรา(Target camera)

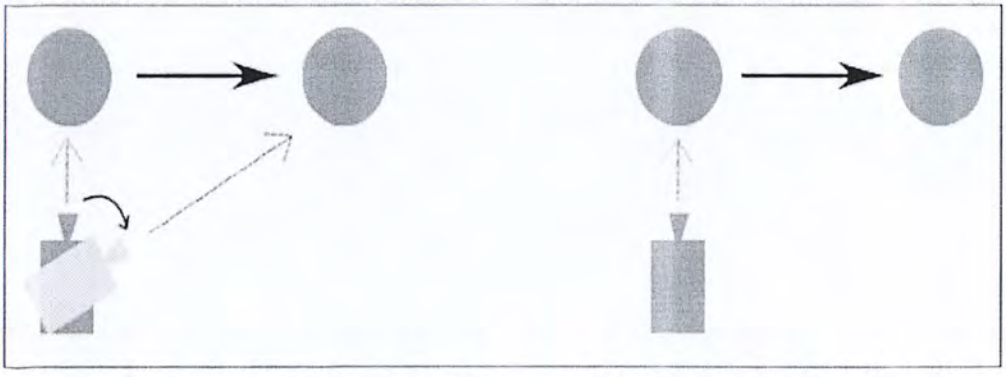
ทาร์เก็ตคาเมราคือประเภทของกล้องที่จะดูที่เป้าหมายตลอดเวลา data type ของทาร์เก็ตเป็น DisplayObject3D เป้าหมายสามารถเป็นที่ว่างหรือจะเป็น primitive วิธีการ define เป้าหมาย ทำได้โดยคำสั่ง

ถ้าผ่านค่า CameraType.TARGET เป็น camera type แล้วยังไม่ได้กำหนดเป้าหมาย Papervision3D จะกำหนดค่าหนึ่งให้โดยใช้คำสั่ง DisplayObject3D.ZERO เป็นการสร้างตำแหน่งว่างๆที่ตำแหน่งตรงกลางของระบบแกน 3 มิติ ผลลัพธ์จะเป็นว่ากล้องจะดูอยู่ที่จุด origin(0,0,0) ของ scene ตลอดเวลา

4.2.6.2.2 ฟรีคาเมรา(free camera)

ฟรีคาเมราทำงานเหมือนกับทาร์เก็ตคาเมรายกเว้นว่าจะไม่มีเป้าหมาย (target) ฟรีคาเมรามักจะมองตรงไปข้างหน้าในทิศทางของของแกน z ถ้าตั้งค่า CameraType.FREE เป็นชนิดของอาร์กิวเมนต์ทาร์เก็ตจะ return ค่าเป็น null

ภาพประกอบแสดงให้เห็นถึงความแตกต่างระหว่างกล้องสองประเภท ภาพประกอบทางด้านซ้ายแสดงให้เห็นกล้องกำหนดเป้าหมาย(target camera)ที่เล็งไปตามวัตถุเคลื่อนตัวจากซ้ายไปขวา เป็นวัตถุที่ได้รับตั้งเป็นเป้าหมายของกล้องและ ในภาพประกอบที่ด้านขวาจะเห็นกล้อง(free camera)ที่ไม่หันไปตามวัตถุที่เคลื่อนที่



รูป 4.26 ความแตกต่างระหว่างกล้องแบบทาร์เก็ตคาเมราและฟรีคาเมรา

4.2.6.2.3 The debug camera

การตั้งค่าต่างๆที่มีในครั้งแรกจะเปลี่ยนไปค่อยข้างง่ายในขณะที่กำลังสร้างapplication 3 มิติ เพื่อการติดตามการตั้งค่าของกล้องที่เปลี่ยนไปในขณะทดสอบ the debug camera จึงถูกพัฒนาขึ้น the debug camera มีส่วนของ built-in navigation มีการตอบสนองต่อการใช้เมาส์และปุ่ม key ต่างๆ แล้วจะแสดงค่าต่างๆเพื่อบอกสิ่งที่เกิดขึ้นอย่างต่อเนื่อง

ค่าต่างๆที่ The debug camera แสดงขณะ debug

1) Coordinates
2) Rotation
3) Field of view
4) Near and far

4.2.6.2.4 The spring camera

spring camera เป็นประเภทกล้องที่ติดตามวัตถุ 3 มิติ เสมือนมีบุคคลที่สามหรือกล้องคอยไล่ตามเหมือนที่เห็นในหลายๆเกม 3 มิติ

4.2.6.3 The Camera3D object

สร้างคลาสด็อคคิวเม้นใหม่ที่ขยาย Chapter03SampleBase ด้วยโค้ดนี้ แทนที่รายละเอียดการทำงานของคลาส _createCamera () ในการตรวจสอบคุณสมบัติของกล้องต่างๆ ที่มีสำหรับจัดการวัตถุ Camera3D ดังตัวอย่าง 4.36

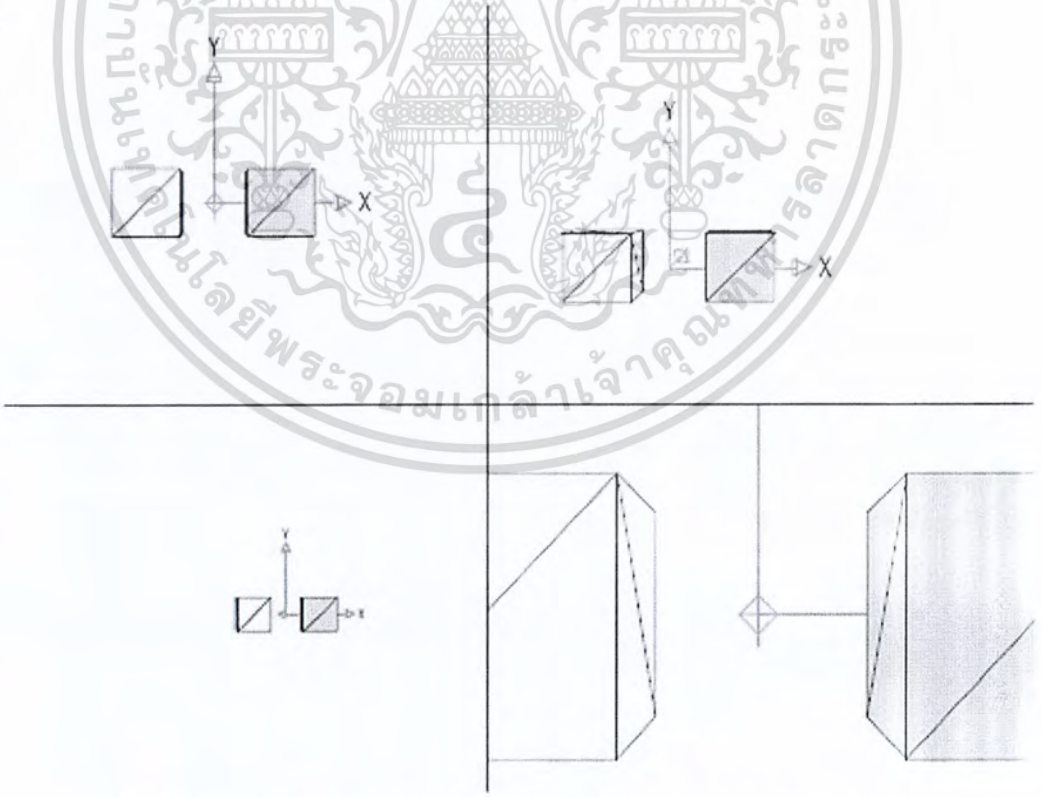
ตัวอย่าง 4.36

```
package flash3dbook.ch03
    {import away3d.cameras.*;
    [SWF(width="800", height="600")]
    public class CameraProperties extends Chapter03SampleBase
        {protected override function _createCamera() : void
            {// Create a new camera object
            var camera : Camera3D = new Camera3D();
            camera.x = 0;
            camera.y = 0;
            camera.z = -1000;
            //Assign the new camera to the view
            _view.camera = camera;  } } }
```

ในโค้ดก่อนหน้านี้ ทำการแทนที่กล้องเริ่มต้นด้วยการตั้งค่าคุณสมบัติในมุมมองกล้อง ด้วยค่าใหม่ของ Camera3D สร้างค่าเริ่มต้นของกล้องที่ตำแหน่งมุมมอง และเนื่องจากได้ใช้อินสแตนซ์ Camera3D มีการควบคุมมากกว่าการเคลื่อนไหวที่กล้องสามารถดำเนินการได้ โดยสามารถปรับคุณสมบัติ Camera3D กับการตั้งค่าใหม่ โดยจะให้ผลลัพธ์ในมุมมองที่แตกต่างกัน

4.2.6.3.1 การย้ายกล้อง

คอมไพล์คลาส CameraProperties ในขั้นตอนนี้จะแสดงในสิ่งที่ได้เห็นแล้วในก่อนหน้านี้ ตัวอย่างที่แสดงในภาพบนด้านซ้ายของรูปที่ 4.27 ในการตั้งค่านี้มีความสามารถในการเปลี่ยน x y และ z ซึ่งเป็นคุณสมบัติของกล้องโดยตรงควบคุม X Y Z และส่วนประกอบของตำแหน่งเวกเตอร์ในพื้นที่ 3 มิติของฉาก การเปลี่ยนตำแหน่งกล้องจะส่งผลกระทบต่อการที่ทำให้มุมมองที่สามารถมองเห็นด้วยการตั้งค่าที่สี่แบบแตกต่างกัน แสดงในภาพทั้งสี่ของรูปที่ 4.27 แสดงภาพจากฉากเดียวกันโดยกล้องสี่ตำแหน่งที่แตกต่างกัน ภาพบนด้านซ้ายจะใช้ค่าเริ่มต้นตำแหน่งกล้อง x = 0 y = 0 z = -1.000 ในภาพขวาบน กล้องจะถูกย้ายขึ้นไปและพิกัดเริ่มต้นคือ x = 150 y = 100 z = -1,000 ในภาพด้านล่างซ้ายก็จะถูกย้ายไปทางซ้าย x = -200 y = 0 z = -2,000 และในภาพด้านล่างขวาจะเดินหน้าไปยัง x = 0 y = 0 z = -200



รูป 4.27 เปรียบเทียบกล้องในฉากเดียวกันจากสี่ตำแหน่งกล้องที่แตกต่างกัน

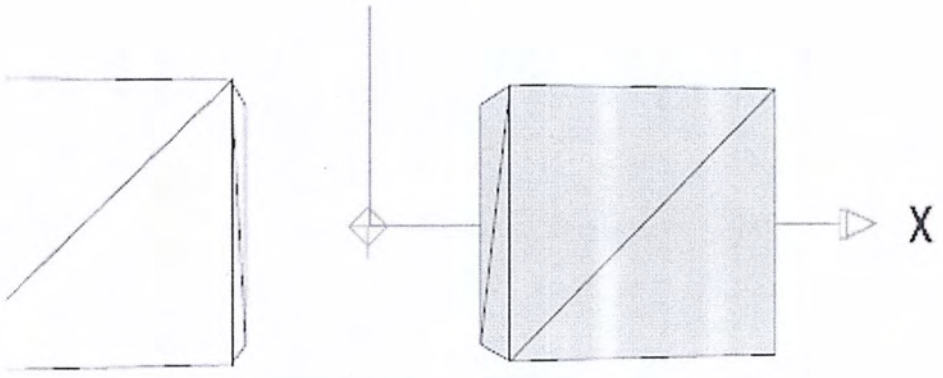
4.2.6.3.2 การหมุนกล้อง

กล้องในอเวย์ทรีดีหมุนโดยใช้ rotationX rotationY และ rotationZ เป็นคุณสมบัติที่วัตถุ 3 มิติใช้ในงานปกติ ชื่อของคุณสมบัติแต่ละแบบสะท้อนให้เห็นถึงการหมุนรอบแกนที่จะดำเนินการ และค่าในคุณสมบัตินี้จะเป็นองศาในการหมุน อย่างไรก็ตามผลของการหมุนกล้องจะแตกต่างจากการหมุนวัตถุ เพราะกล้องรับหน้าที่เป็นมุมมอง ซึ่งหมายความว่าการหมุนรอบแกน X ผลในการเคลื่อนไหวคล้ายกับการก้มศรีษะ หมุนรอบแกน Y คล้ายกับการสั่นหัวของคุณบอกว่า"ไม่"และหมุนรอบแกน Z คล้ายกับการเอนหัวไปด้านซ้ายหรือขวา เนื่องจากอาจมีน้ำอยู่ในหู ผลของการหมุนกล้องอย่างง่าย โดยการแทนที่ _createCamera ()ในตัวอย่าง CameraProperties ด้วยโค้ดดังตัวอย่าง 4.37

ตัวอย่าง 4.37

```
// Create a new camera object
var camera : Camera3D = new Camera3D();
camera.x = 0;
camera.y = 0;
camera.z = -1000;
// Rotate the camera by 10 degrees around the Y-axis
camera.rotationY = 10;
//Assign the new camera to the view
_view.camera = camera;
```

ตอนนี้จะตั้งค่าตำแหน่งกล้อง เพื่อตั้งค่าเริ่มต้นจากมุมมอง จากนั้นหมุนกล้อง 10 องศารอบแกน Y ทำการคอมไพล์ CameraProperties จะแสดงผลแสดงในรูปที่ 4.28 ด้วยมุมมองที่ปรากฏเหมือนกับว่าคุณได้หันศรีษะของคุณไป 10 องศาทางขวา ลูกลูกบาศก์ทางซ้าย (สีขาว) จะไปปรากฏในพื้นที่แสดง และลูกบาศก์ทางขวา (สีเทา) จะปรากฏขึ้นเล็กน้อยทางด้านซ้ายของจุดกึ่งกลางพื้นที่แสดง ในภาพตัวอย่างคุณสมบัติของกล้อง จะแสดงหลังจากได้ทำการหมุนกล้องไป 10 องศารอบๆแกน Y และหมุนรอบๆพื้นผิวที่แสดงไปทางด้านขวา



รูปที่ 4.28 ตัวอย่างหลังจากได้ทำการหมุนกล้องไป 10 องศา

4.2.6.3.3 การปรับค่าการซูม และคุณสมบัติการโฟกัส

จากที่ได้กล่าวไว้ข้างต้นแล้ว กล้องในอเวย์ทรีดีจะมีคุณสมบัติที่จะสามารถให้ทำการปรับค่ารูปภาพได้คล้ายคลึงกับกล้องจริง สำหรับการจัดเรียงลำดับของแอปปลิเคชัน จะไม่พูดถึงจุดเปิดรับแสง และการตั้งค่าของความเร็วชัตเตอร์ แต่มีเพียงไม่กี่คุณสมบัติทางด้านเสียงที่คล้ายคลึงกันกับ Camera3D เช่น การซูม โฟกัส และเลนส์ ต่อไปจะศึกษาเกี่ยวกับผลกระทบของการปรับค่าในสองคุณสมบัติที่ได้พูดถึงในข้างต้นแล้ว การควบคุมเลนส์จะมีความซับซ้อนเล็กน้อย

การพิจารณาผลกระทบของการซูม และโฟกัส สามารถเปลี่ยนโค้ดของ _createCamera ใน CameraProperties ด้วยโค้ดต่อไปนี้ ดังตัวอย่าง 4.38

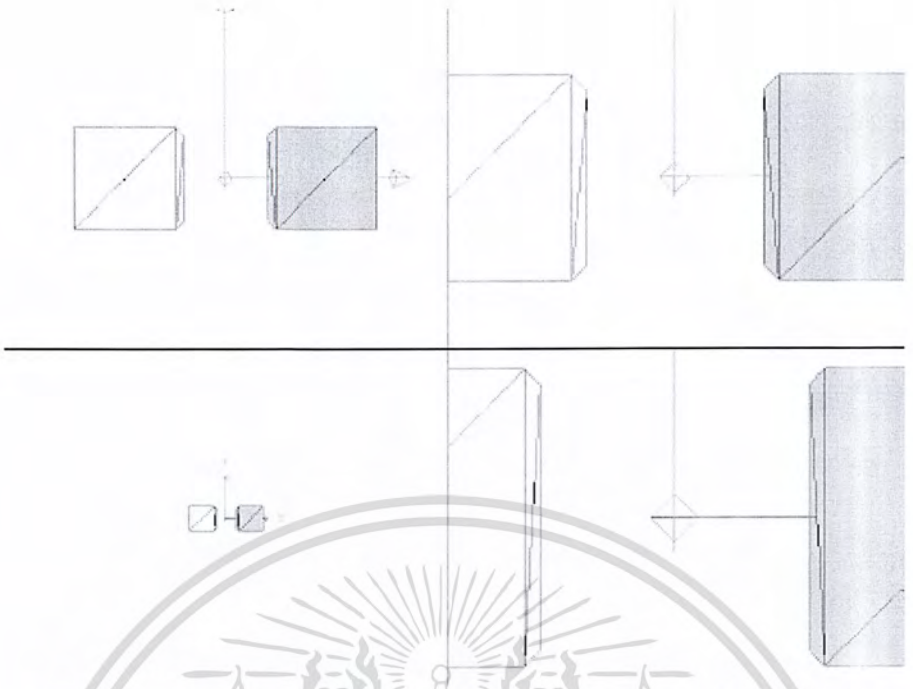
ตัวอย่าง 4.38

```
// Create a new camera object
var camera : Camera3D = new Camera3D();
camera.x = 0;
camera.y = 0;
camera.z = -500;
//set the zoom and focus properties
camera.zoom = 10;
camera.focus = 100;
//Assign the new camera to the view
_view.camera = camera;
```

ต่อไปจะทำการตั้งค่าตำแหน่งกล้องไปที่ -500 ในแกน Z ซึ่งจะเป็นครึ่งหนึ่งของระยะทางเริ่มต้นจากจุดกึ่งกลางของฉาก รวมถึงจะทำการสร้างการตั้งค่าเริ่มต้นของการซูม และโฟกัสที่ 10 และ 100 ตามลำดับ ดังนั้น ภายหลังการคอมไพล์ จะสามารถปรับค่าดังกล่าว ตัวอย่างการคอมไพล์ CameraProperties อีกครั้งจะแสดงดังภาพทางซ้ายบนของรูปที่ 4.29

ต่อไปจะเป็นการเริ่มปรับค่า โดยการตั้งค่าการซูมที่ 20 หรือสองเท่าของค่าเริ่มต้นนั้น ผลกระทบของมุมมองมีมากเท่ากับที่กับผลกระทบที่จะคาดถึงในการควบคุมการซูมของกล้องสภาวะจริง จะแสดงผลลัพธ์ตามสัดส่วนการซูม คอมไพล์ตัวอย่าง CameraProperties จะได้ผลลัพธ์ที่ปรากฏในมุมขวาบนของรูปที่ 4.29โดยมีฉากเป็นสองเท่าด้วยการตั้งค่านี้

จากนี้จะทำการพิจารณาผลกระทบของการปรับค่าโฟกัสในกรณีนี้ ค่าของโฟกัสจะไม่สัมพันธ์กันกับการตั้งค่าโฟกัสในสภาวะจริง แต่จะสามารถแสดงให้เห็นระยะทางระหว่างตำแหน่งของกล้องและมุมมองบนระนาบ ในงาน 3 มิติ กราฟฟิก ระนาบมุมมองคือพื้นผิวที่มองไม่สามารถมองเห็นในพื้นที่ที่เป็นพื้นฐานของฉากฉายไปยังมุมมอง ถ้าจินตนาการหน้าจอคอมพิวเตอร์คือระนาบมุมมอง โฟกัสจะสามารถปรับระยะทางตามทฤษฎีระหว่างพื้นผิวของระนาบนี้และตำแหน่งกล้องที่อยู่ด้านหน้าของระนาบ ผลของการปรับโฟกัสในกล้องจะใกล้เคียงกับมุมมองของระนาบ รวมถึงมุมกว้างสุดของมุมมอง ถ้าทำการปรับค่าโฟกัสไปที่ 10 และคอมไพล์ CameraProperties ผลลัพธ์ (แสดงในภาพด้านล่างซ้ายของรูปที่ 4.29) ผลลัพธ์จะคล้ายกับเลนส์มุมกว้างรวมกับฉากที่มากขึ้น รวมถึงในวิวพอร์ตด้วยมุมมองสูงที่มีการบิดเบือน ค่าของจุดโฟกัสจุดใหญ่จะวางในกล้องเพิ่มเติมจากมุมมองของระนาบ ผลลัพธ์ที่ได้จะแคบลงกว่าพื้นที่ของมุมมองและสูญเสียมุมมองที่มีการบิดเบือน การตั้งค่าโฟกัสไปที่ 500 และซูมกลับไปที่ 10 ทำการคอมไพล์ ผลลัพธ์จะแสดงให้เห็นภาพในมุมขวาด้านล่างของรูปที่ 4.29 ด้วยมุมมองที่มีฉากขนาดใหญ่ โดยแสดงเฉพาะด้านข้างของลูกบาศก์ จากภาพจะแสดงการปรับการตั้งค่าการซูมและโฟกัส ในด้านบนซ้ายภาพคุณสมบัติกล้องถูกตั้งไว้เป็นค่าเริ่มต้นของการซูม = 10 และโฟกัส = 100 ในด้านขวาบนภาพจะถูกตั้งค่าคุณสมบัติของกล้องเพื่อซูม = 20 และโฟกัส = 100 ในภาพด้านล่างซ้ายคุณสมบัติกล้องถ่ายรูปเพื่อซูม = 20 และโฟกัส = 10 และในภาพด้านล่างขวามีการกำหนดคุณสมบัติของกล้องเพื่อซูม = 10 และโฟกัส = 500



รูป 4.29 เปรียบเทียบฉากเดียวกันกับการตั้งค่าการซูมและโฟกัสที่แตกต่างกันในกล้อง

ตัวอย่างก่อนหน้านี้ได้แสดงให้เห็นจุดโฟกัสและการซูมของกล้อง จะอนุญาติให้ปรับขนาดของมุมมอง และปรับขนาดของมุมมองที่มีการบิดเบือน และปรับมุมมองที่นำมาใช้เมื่อฉายฉากในมุมมอง คุณสมบัติเหล่านี้จะใช้ได้กับทุกสภาพกล้องในอเว่ย์ทรีดี

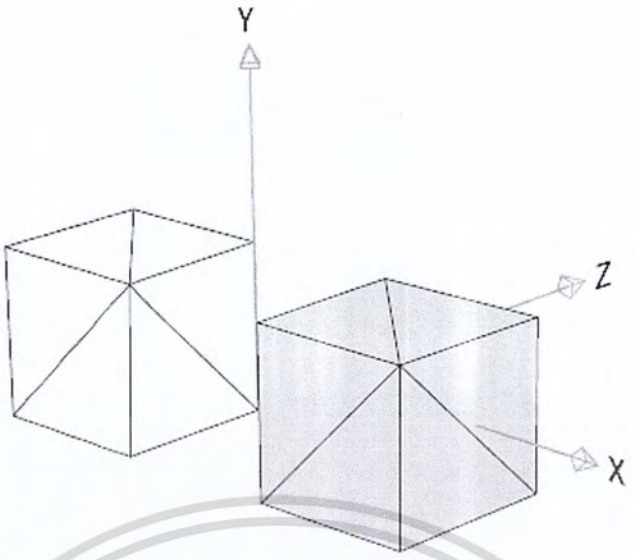
4.2.6.3.4 การใช้งาน lookAt ()

บ่อยครั้งเหตุผลของการหมุนกล้อง คือ การนำจุดกึ่งกลางของวิวพอร์ตมาวางบนจุดใดจุดหนึ่งของพื้นที่ ดังตัวอย่างเช่น วัตถุที่น่าสนใจ การคำนวณงานที่แม่นยำจำเป็นต้องไว้ที่จุดกึ่งกลางของวัตถุซึ่งเป็นงานที่ค่อนข้างวุ่นวาย จึงเป็นเหตุผลที่อเว่ย์ทรีดีมีวิธีการที่จะให้ประสบความเร็จโดยเรียกว่า lookAt () ซึ่งบ่งบอกอยู่ในคลาสของกล้อง วิธีการของ lookAt () ต้องการตำแน่งของเวกเตอร์ประเภท away3d.math.Number3D (3 มิติของตัวแปรคลาส Point พื้นฐานใน แฟลช) นี้จะใช้สำหรับตำแหน่งในพื้นที่ซึ่งเกี่ยวกับกล้องที่ใช้ในการหมุน ต่อไปจะเป็นการสร้างตัวอย่างซึ่งขยายคลาสตัวอย่าง Chapter03SampleBase ด้วยโค้ดต่อไปนี้ เพื่อพิจารณาวิธีการทำงานของ lookAt ดังตัวอย่าง 4.39

ตัวอย่าง 4.39

```
package flash3dbook.ch03
    {
     import away3d.cameras.*;
     import away3d.core.math.Number3D;
     [SWF(width="800", height="600")]
     public class CameraLookAt extends Chapter03SampleBase
      {protected override function _createCamera() : void
            {
             // Create a new camera object
             var camera : Camera3D = new Camera3D();
             camera.x = 1000;
             camera.y = 500;
             camera.z = -1000;
             //Use lookAt() to point the camera towards the center of the scene
             camera.lookAt(new Number3D(0, 0, 0));
             //Assign the new camera to the view
             _view.camera = camera;
             }
        }
    }
```

ในวิธีการของ _createCamera() วัตถุใหม่ใน Camera3D สร้างด้วยตำแหน่งเวกเตอร์ที่ตั้งค่าไว้ที่ (1000,500, –1000) ในพื้นที่ (ด้านบนและด้านขวา) ต่อจากนั้นใช้ lookat() ในกล้อง เพื่อกำหนดจุดไปทางจุดกำเนิดของพื้นที่ที่ (0, 0, 0) การคอมไพล์ของโค้ด จะแสดงดังรูป 4.30 ตัวอย่าง หลังจากหมุนกล้องไปด้านบน และด้านขวา โดยใช้ lookAt() เพื่อกำหนดจุดไปที่จุดกำเนิดของพื้นที่



รูป 4.30 หลังจากหมุนกล้องไปด้านบน และด้านขวา โดยใช้ lookAt()

บ่อยครั้ง ผู้ใช้ต้องการจุดกึ่งกลางของวิวพอร์ต มากกว่าจุดใดๆในพื้นที่ วัตถุ 3 มิติ มีตำแหน่งแน่นอน เพื่อที่จะแสดงตำแหน่งเวกเตอร์ของวัตถุตามพิกัด เรียกว่า Number3D คุณสมบัตินี้สามารถเรียกใช้ด้วยวิธีการเดียวกับ lookAt() ดังตัวอย่างที่ผ่านมา เพื่อที่จะบ่งบอกกล้องไปทางตำแหน่งของวัตถุ ลองปฏิบัติด้วยการนำคำสั่ง lookAt() ไปแทน CameraLookAt ด้วยโค้ดต่อไปนี้ที่ตรงกลางของวิวพอร์ตบนลูกบาศก์สีเทาในของเดิม ซึ่งวางอยู่บนตำแหน่งทางด้านขวาของพื้นที่ ดังตัวอย่าง 4.40

ตัวอย่าง 4.40

```
//Use lookAt() to point the camera towards the cube2 primitve
camera.lookAt(_cube2.position);
```

วิธีการของ lookAt() บนวัตถุ Camera3D อ้างอิงมาจากคลาส Object3D ซึ่งหมายความว่า ไม่ว่าวัตถุ 3D ใดๆ สามารถหมุนไปทางตำแหน่งของเวกเตอร์ในพื้นที่ตามจุด อย่างไรก็ตามในทางปฏิบัติเป็นที่รู้กันดีอยู่ว่าวิธีการนี้ใช้สำหรับการหมุนกล้อง

4.2.6.4 The TargetCamera3D object

ส่วนมากการใช้ประโยชน์ของอเวย์ทรีดี ที่สร้างขึ้นนั้น จะเป็นภาพเคลื่อนไหว และสามารถโต้ตอบได้มากกว่าที่จะเป็นภาพนิ่ง ที่จุดในจุดหนึ่งอาจจะดูเหมือนว่ามีกล้องเคลื่อนที่

ติดตามวัตถุ ดังนั้นวัตถุจึงสามารถอยู่ในมุมมองตลอดเวลา ถ้าได้ทำตามตัวอย่างข้างต้น จะสามารถจะจัดการกับปัญหาที่ดีที่จะใช้วิธีการ lookAt () ในการแก้ปัญหาได้ในทุกๆเฟรม อาจจะเป็นไปได้ว่าตั้งแต่ติดตามวัตถุเป็นความสามารถโดยทั่วไปของอเวย์ทรีดีวัตถุ TargetCamera3D จะทำเช่นนี้ให้โดยอัตโนมัติ คุณสมบัติของตำแหน่งสามารถบ่งบอกวัตถุ

TargetCamera3D ซึ่งวัตถุ 3 มิติ ที่ติดตาม และการหมุนของกล้องจะปรับปรุงโดยอัตโนมัติในแต่ละครั้งที่ผู้ใช้สร้างมุมมองขึ้นมาใหม่

สร้างตัวอย่าง Chapter03SampleBase และตั้งค่ากล้องเป้าหมาย เพื่อจะติดตามวัตถุ ในขณะที่กล้องมีการเคลื่อนที่ ดังตัวอย่าง 4.41

ตัวอย่าง 4.41

```
package flash3dbook.ch03
    {import away3d.cameras.*;
     import flash.events.Event;
     [SWF(width="800", height="600")]
     public class TargetCameraMovement extends Chapter03SampleBase
        {protected override function _createCamera() : void
            {// Create a new camera object
            var camera : TargetCamera3D = new TargetCamera3D();
            camera.z = -1000;
            //Assign the camera target as cube2
            camera.target = _cube2;
            //Assign the new camera to the view
            _view.camera = camera;  } } }
```

กล้องเป้าหมายในโค้ดก่อนหน้านี้ได้รับการกำหนดค่าที่จะกำหนดเป้าหมาย cube2 ในฉาก ซึ่งลูกบาศก์สีเทามองเห็นได้ทางด้านขวาของจุดเริ่มต้นในฉาก การคอมไพล์ตัวอย่างจะแสดงมุมมองด้วยวัตถุ cube2 ที่จุดกึ่งกลางของตัวเอง

ในการแนะนำตัวอย่างข้างต้นที่เห็นใน CameraLookAt ที่ผ่านมาจะแสดงให้เห็นถึงวิธีการ lookAt โดยจะมีการยืนยันการเปลี่ยนแปลงในทุกๆเฟรม ใน TargetCamera3D โดยใช้ _onEnterFrame เพื่อให้แสดงเป็นภาพเคลื่อนไหว โดยเพิ่มไว้ในบรรทัดสุดท้ายของคลาส TargetCameraMovement ดังตัวอย่าง 4.42

ตัวอย่าง 4.42

```
protected override function _onEnterFrame(ev : Event) : void
{ _view.camera.y = -(stage.mouseY - stage.stageHeight/2);
 _view.camera.x = stage.mouseX - stage.stageWidth/2;
 _view.render(); }
```

ทำการคอมไพล์ตัวอย่าง TargetCameraMovement จะเห็นว่าจุดพิกัด x และ Y ของกล้องจะปรับปรุงตามพิกัดของ X และ Y ตามเมาส์ ในทุกๆตำแหน่ง และกล้องจะทำการเก็บวัตถุ cube2 และระบุตำแหน่งในจุดกึ่งกลางของมุมมอง

4.2.6.5 The HoverCamera3D object

โปรแกรม 3 มิติ ที่พบบ่อยในแฟลชเป็นสิ่งหนึ่งในการช่วยให้ผู้ใช้สามารถดูวัตถุ 3 มิติ ได้ในทุกด้าน โดยหมุนกล้องไปรอบๆ ด้วยการใช้เมาส์ สำหรับการทำงานของคลาส HoverCamera3D เป็นกล้องในอุดมคติช่วยลดความยุ่งยากในการหมุนและตำแหน่ง เพื่อที่จะต้องการหมุนรอบวัตถุ ด้วยคุณสมบัติพิเศษ

คลาส HoverCamera3D สืบทอดมาจากคลาส TargetCamera3D และได้เพิ่มคุณสมบัติบางอย่างเพื่อที่จะเติมเต็มคุณสมบัติ เพิ่มโค้ดลงในคลาส ดังตัวอย่าง 4.43

ตัวอย่าง 4.43

```
package flash3dbook.ch03
    {import away3d.cameras.*;
     import flash.events.Event;
     [SWF(width="800", height="600")]
     public class HoverCameraMovement extends Chapter03SampleBase
{private var _hoverCamera : HoverCamera3D;
        protected override function _createCamera() : void
            {// Create a new camera object
             _hoverCamera = new HoverCamera3D();
             _hoverCamera.distance = 2000;
             _hoverCamera.tiltAngle = 10;
             _view.camera = _hoverCamera; }      //Assign the new camera to the view
```

ตัวอย่าง 4.43 (ต่อ)

```
        protected override function _onEnterFrame(ev : Event) : void
            {_hoverCamera.panAngle = stage.mouseX -
stage.stageWidth/2;
             _hoverCamera.hover();
             _view.render(); }
            }
        }
```

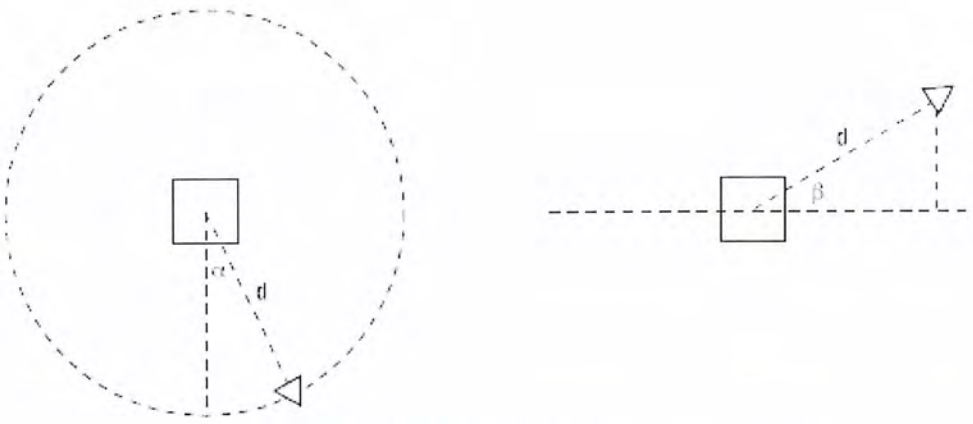
วิธีการ _createCamera() ในโค้ดก่อนหน้านี้ สามารถสร้าง HoverCamera3D และตั้งค่าคุณสมบัติสองอย่าง ซึ่งเรียกว่า distance และ tiltAngle ต่อเนื่องมาจากวิธี _onEnterFrame() วัตถุ HoverCamera3D จะมี _panAngle ในการกำหนดค่าในแกน X ของเมาส์ การคอมไพล์ควรดูที่พิกัด X ของเมาส์ที่ควบคุมการหมุนในแนวนอน ซึ่งกล้องจะฉายมาที่มุมสูงของลูกบาศก์

เพื่ออธิบายสิ่งที่เกิดขึ้น จำเป็นต้องวิเคราะห์สามคุณสมบัติที่เพิ่มเข้ามาใหม่ในกล้อง คือคุณสมบัติ distance panAngle และ tiltAngle ใน HoverCamera3D

ต่อไปจะพูดถึงคุณสมบัติ panAngle และ tiltAngle ของการหมุน (องศา) ดำเนินการเลื่อนกล้องโดยขึ้นอยู่กับ $\propto$ และ β ตามลำดับ ลูกบาศก์และกล้องถ่ายภาพจะปรากฏขึ้นทางด้านบนจากภาพด้านซ้าย และจากด้านขวาในภาพ การเพิ่มขึ้นหรือลดลงของค่า panAngle ซึ่งมีสาเหตุมาจากการหมุนกล้องรอบวัตถุเป้าหมายในระนาบแนวนอนดังภาพด้านซ้าย

การเพิ่มขึ้นหรือลดลงของค่า tiltAngle ซึ่งมีสาเหตุมาจากมุมเงยแทนดังภาพทางด้านขวา การกำหนดระยะทั้งสองของการหมุน ทั้งมุมและรัศมีจะใช้ในการกำหนดตำแหน่ง โดยลักษณะแบบนี้เรียกว่าเป็นขั้วพิกัด

ภาพนี้แสดงมุมมองของฉากโดยใช้กล้องที่ขึ้นอยู่กับเมาส์ ภาพซ้ายแสดงถึงคุณสมบัติ panAngle ภาพขวาแสดงให้เห็นฉากที่แสดงจากด้านข้าง โดย β แสดงถึงคุณสมบัติ tiltAngle ของกล้อง โปรดทราบว่า d หมายถึงระยะทาง ในทั้งสองกรณีนี้



รูปที่ 4.31 คุณสมบัติ panAngle และ tiltAngle

ในตัวอย่าง tiltAngle ถูกตั้งค่าเป็นค่าคงที่ 10 องศาให้กล้องสูงขึ้นเล็กน้อยในมุมมองของฉาก และpanAngle ถูกตั้งค่าให้กับสามารถปรับตำแหน่งตามเมาส์ให้ผู้ใช้สามารถควบคุมในแนวนอน ตำแหน่งที่เกิดขึ้นจริงและค่าการหมุนของวัตถุ HoverCamera3D มีการคำนวณใหม่ในทุกเฟรมจาก tiltAngle panAngle และระยะทาง เรียกคุณสมบัตินี้ว่า hover () เป็นวิธีการควมคุมมุมองของกล้องโดยใช้เมาส์ใน _onEnterFrame () ก่อนที่จะมีการแสดงผล

อาจพบว่ามีตัวอย่างมีการทำงานที่ช้าลง เมื่อนำไปใช้กับการเคลื่อนไหวเมาส์ที่สัมพันธ์กับกล้อง ทั้งนี้เนื่องจากการปรับปรุงพิกัดใน HoverCamera3D ตามค่าจริงในตำแหน่งใหม่ที่เมาส์เลื่อนไป ดังนั้นอาจแก้ไขได้โดยตั้งค่าเริ่มต้นให้มีค่าเท่ากับ 8 โดยใช้โค้ดบรรทัดนี้เพิ่มเข้าไปใน _createCamera () ซึ่งอยู่ในคลาส HoverCameraMovement ดังตัวอย่าง 4.44

ตัวอย่าง 4.44

```
hoverCamera.steps = 0;
```

กำหนดค่าให้เป็น 0 ทำให้การเคลื่อนไหวของกล้องไม่มีปัญหาการติดขัด และคอมไพล์ตัวอย่างจะเห็นว่ามีโปรแกรมมีการทำงานที่เร็วขึ้น หรืออีกวิธีหนึ่งให้ตั้งค่าเป็น 16 จะมีคุณสมบัติที่ให้ผลตรงข้าม และการเคลื่อนไหวของกล้องจะราบรื่นเมื่อเมาส์มีการเคลื่อนที่

4.3 พื้นฐานของโมเดล และ Sprites

ในส่วนมากอเวย์ทรีดีโปรเจค Scene สามารถแบ่งออกเป็นสามประเภทใหญ่ๆได้แก่ Primitives คือ รูปทรงเรขาคณิตสามมิติแบบธรรมดา (อย่างเช่นลูกบาศก์ที่ได้แสดงดังในตัวอย่างในบทที่แล้ว)

ถูกสร้างขึ้นจากภายในโดย the engine จากการรวบรวมของพรอเพอร์ตีที่ได้ตั้งไว้ล่วงหน้าแล้ว Models สร้างขึ้นมาจากรูปทรงเรขาคณิตจากการนำเข้าของไฟล์ประเภท 3 มิติ อย่างเช่น .dae .3ds หรือ .obj ที่สร้างขึ้นมาจากการใช้ 3D modeling software Sprites ในสามมิติ คือ flat images ที่ประกอบด้วยขนาดและระยะทางโดยไม่สนใจการหมุนของรูป ถึงแม้ว่าจะหันไปทางกล้องถ่ายรูปแล้วอย่างแน่นอน

ไม่จำเป็นต้องรู้ทุกเรื่องเกี่ยวกับความแตกต่างของประเภทในการสร้าง High-quality 3D แต่ความรู้นั่นแหละเป็นสิ่งที่สามารถสรรสร้างความคิดต่างๆได้ ตัวอย่างในการปฏิบัติมีอยู่ท้ายบท โดยจำลองว่า the topics สามาถใช้ร่วมกับ maximum effect ในงาน single project.

4.3.1 คำศัพท์เฉพาะ

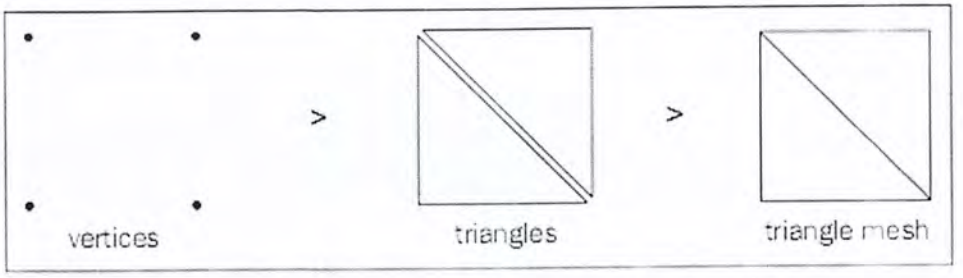
Primitive เป็นคำทั่วไปที่ใช้ในงานกราฟฟิค 3 มิติรวมถึงอเว์ยทรีดีโดยอธิบายถึงรูปทรงเรขาคณิตสามมิติแบบธรรมดา อย่างเช่น ลูกบาศ์ก รูปทรงกลม รูปทรงกระบอก รวมถึงใช้ในเวลาที่จะพูดถึงวัตถุ 3 มิติ อย่างเช่น รูปใบหน้า หรือ line segment แต่ในอเว์ยทรีดี จะอ้างอิงถึงว่าคือ elements. Elements คือ พื้นผิวที่สามารถเห็นได้ หรือว่า เส้นที่แสดงในสามมิติ และมีรูปแบบที่กำหนดโดยข้อมูลพื้นฐานทั้งหมดของสามมิติ

4.3.1.1 Vertices

ในเรขาคณิตสามมิติ วัตถุจะประกอบด้วยกลุ่มของ vertices หรือที่เรียกว่า verts (เอกพจน์จะถูกเรียกว่า vertex) vertex เป็นจุดในพื้นที่สามมิติ เมื่อกำหนดตำแหน่ง sprite ในแฟลช ทั้งสองแกน คือแกน x และแกน y vertex มีแกนที่สามที่ตั้งบนแกน z การจัดลำดับของพิกัดในสามมิติ จึงเป็น x, y, z เสมอ ใช้กับรูปทรงที่มีขนาดใหญ่ โดยไม่สามารถมองเห็นได้ด้วยตัวเอง แต่องค์ประกอบที่สามารถมองเห็นได้ถูกสร้างด้วยกลุ่มของจุดยอดที่แตกต่างกันไป ดังตัวอย่าง สองจุดยอดที่แสดงบนเส้น สามจุดบอดที่แสดงบนสามเหลี่ยม และอีกมากมาย เมื่อกำหนดพื้นผิวบนระนาบสามมิติ รูปสามเหลี่ยมเป็นที่ใช้กันอย่างแพร่หลาย ในอเว์ยทรีดีเส้น และพิ้นผิวจะถูกแสดงด้วย classes Segment and Face ตามลำดับ ในขณะที่จุดยอดจะถูกแสดงด้วยกลุ่มของจุดยอดทั้งหมดสามารถพบเจอได้ใน away3d.core.base package

4.3.1.2 Faces and segments

จุดที่รวมกันเป็นรูปสามเหลี่ยมที่เรียกว่า faces หรือ triangle faces ทุกๆรูปสามเหลี่ยมเกิดจากจุดสามจุด โปรแกรมสามมิติอื่นๆ อาจช่วยให้สามารถสร้าง faces ที่สร้างขึ้นจากจุดสี่จุดซึ่งเรียกว่า quad faces แต่ อเว์ยทรีดี รู้จักเพียงแค่ face สามเหลี่ยมเท่านั้น จุดจะถูกกำหนดเป็นรูปร่างของรูปสามเหลี่ยม และเนื่องจากมีแกนพิกัด x y และ z ทำให้สามารถสร้างสามเหลี่ยมได้อิสระตามแกนสามมิติ เมื่อสามารถวาดรูปสามเหลี่ยมได้หลายๆตำแหน่งกันในพื้นที่ 3 มิติ จึงสามารถสร้างรูปร่างได้ทุกประเภท เช่นในอเว์ยทรีดี ที่เรียกว่า triangle mesh(ตาข่ายสามเหลี่ยม) โดยการสร้าง mesh เพื่อสร้างวัตถุ 3 มิติ



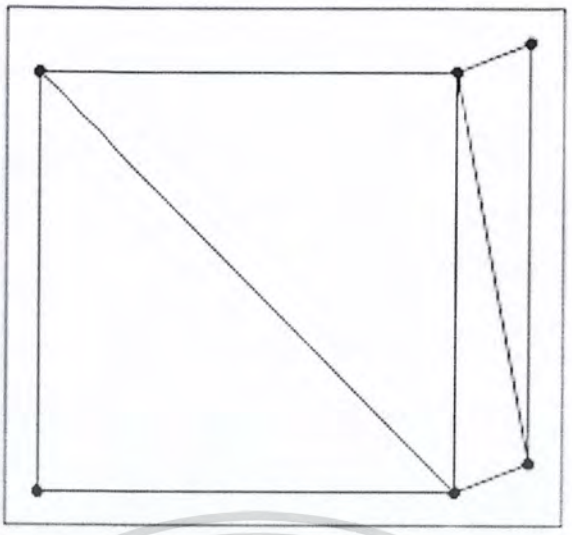
รูป 4.32 อธิบายการเกิด faces สามเหลี่ยมและตาข่ายสามเหลี่ยมจากจุด

Faces and segments เป็นองค์ประกอบที่สามารถมองเห็นได้ในอเวย์ทรีดีและใช้กันอย่างแพร่หลาย ใน 3D applications อื่นๆ พื้นผิวจะประกอบด้วยจุดยอดมากกว่าสามจุด และบางครั้งอาจแสดงถึงรูปทรงหลายเหลี่ยม ในขณะที่พื้นผิวที่มีสามจุดยอดจะเรียกว่า รูปทรงสามเหลี่ยม คำที่กล่าวมาเป็น term ที่ใช้กันอย่างแพร่หลาย แต่ไม่สามารถใช้ได้บ่อยนักในอเวทรีเฟค (Away3D Face) เป็นชื่อเรียกทั่วไปสำหรับพื้นผิว และ segment เป็นชื่อเรียกทั่วไปสำหรับเส้น แต่ละอันสามารถสร้างจากจุดยอดกี่จุดก็ได้ ดังนั้นจึงมี straight edges จำนวนมากมาย ความแตกต่างของ Faces and segments คือ ความสามารถในการสร้าง drawing routines Faces จะถูกวาดโดยพื้นที่ segment จะถูกวาดโดยการเชื่อมเส้น

4.3.1.3 Meshes and primitives

Mesh คือ การรวมของจุดยอดที่ถูกทำให้มองเห็นจากการรวมขององค์ประกอบของจุดยอด อย่างเช่น การ saving measure จุดยอดสามารถนำมาใช้ใหม่โดยองค์ประกอบหลายๆ องค์ประกอบใน mesh มันไม่มีความจำเป็นที่จะสร้างองค์ประกอบด้วยจุดยอดจุดเดียว รูปลูกบาศ์กแบบธรรมดา ตัวอย่างเช่น สามารถระบุตำแหน่งด้วยจุดยอดแปดจุดที่มุมของแต่ละมุม การสร้างวัตถุของแข็ง จุดยอดต่างๆสามารถแบ่งได้ระหว่าง 12 faces โดย หกด้าน ประกอบด้วยรูปทรงสามเหลี่ยมสองรูป รูปลูกบาศ์กเป็นส่วนหนึ่งของวัตถุสามมิติที่เรียกว่า primitives โดยเกิดจากการที่ mesh แต่ละชิ้นส่วนของcode ที่ใช้สมการทางเรขาคณิตเพื่อที่จะสร้าง รูปทรงทางเรขาคณิตของตนเอง ในอเวย์ทรีดีวัตถุ primitives สามารถสร้างได้โดยการใช้รูปแบบที่เจอใน away3d.primitives แต่ละชนิดของ primitives ควบคุมกระบวนการทำงานของการเกิด mesh โดยผ่านทาง กลุ่มคุณสมบัติต่างๆที่ได้รับการปรับปรุงการกำหนดค่า mesh ภายในเมื่อมีการเปลี่ยนแปลง

รูปข้างล่างแสดง mesh สามเหลี่ยมเหมือนในรูปก่อนหน้านี้ แต่มีการขยายเพิ่มโดยเพิ่มคู่สามเหลี่ยมอื่นทำมุมด้วย 90 องศา แต่ละจุดและรูปสามเหลี่ยมทั้งหมดจะถูกวาดเป็นสองมิติ จะให้ภาพลวงตาของวัตถุสามมิติที่มีความกว้างความสูงและความลึก



รูป 4.33 สี่เหลี่ยมถูกวาดสองมิติ แต่แสดงผลลัพธ์เป็นสามมิติ

โดยการตั้งค่าพิกัดของจุด x y และ z สามารถกำหนดให้ฉายภาพเหล่านั้นลงบนหน้าจอสองมิติ เพื่อสร้างภาพลวงตาของความลึกที่กำหนดไว้ หรือพูดอีกอย่างหนึ่งว่า รู้วิธีการแสดงภาพรูปทรงสามมิติมาเป็นภาพสองมิติ

4.3.1.4 Billboards and sprites

บิลบอร์ดเป็นองค์ประกอบ 2 มิติที่สามารถวางไว้ภายในระนาบ 3 มิติได้ ภายในไม่มีข้อมูลเกี่ยวกับการหมุน ดังนั้นเวลาที่แสดงผลต่อหน้ากล้องจะแสดงในรูปของภาพไม่มีมิติที่แสดงด้วยขนาดและระยะทาง เพราะฉะนั้นต้องการแค่จุดยอดจุดเดียวเพื่อแสดงตำแหน่งบนฉาก บิลบอร์ดเป็นส่วนประกอบหนึ่งของคลาส 3 มิติ ซึ่งโดยทั่วไปสามารถเรียกว่าเป็น sprite ถึงแม้ว่าจะมีความแตกต่างพื้นฐานบางอย่างระหว่างบิลบอร์ด และ sprite ซึ่งคุ้ยเคยการอยู่แล้วในแฟลช รูปแบบพื้นฐานสามารถสร้างด้วยคลาส Sprite3D ซึ่งอยู่ในแพคเกจ away3d.sprites

เมื่อก่อน sprites 3 มิติใช้กันอย่างแพร่หลายในเกมส์ 3 มิติ เพราะว่าใช้งานง่าย ในการแสดงให้เห็นถึงรูปทรงเชิงซ้อน โดนไม่ต้องมีขั้นตอนในการสร้างมากจนเกินไป บิลบอร์ดเป็นรูปแบบที่ง่ายที่สุดที่สามารถวางวัตถุด้วยรูปทรงกลมที่สมมาตร (เช่นวัตถุที่มีรูปทรงโค้งมนทุกๆ ด้าน) ตัวอย่างเช่นอนุภาคทรงกลมและก้อนเมฆ sprite ที่มีความซับซ้อนมากสามารถใช้ภาพที่แตกต่างกันสำหรับมุมมองที่แตกต่างเช่นกัน (วิธีการนี้สามารถเจอในเกมส์ต่อสู้สมัยก่อนสำหรับการวาดภาพศัตรู) หรือการหมุนเป็นการแสดงวัตถุด้วยแกนสมมาตร (วัตถุมีผิวราบเรียบเหมือนกับมุมที่จำกัด) ดังเช่นต้นไม้ หรือถังไม้

4.3.2 ตั้งค่าคลาสพื้นฐาน

การตั้งค่าเพื่อสร้างคลาส Chapter04SampleBase ใน flash3dbook.ch04 ดังตัวอย่าง 4.45

ตัวอย่าง 4.45

```
package flash3dbook.ch04
    {import away3d.cameras.*;
     import away3d.containers.*;
     import flash.display.*;
     import flash.events.*;
     public class Chapter04SampleBase extends Sprite
          {protected var _camera : HoverCamera3D;
          protected var _view : View3D;
          public function Chapter04SampleBase()
              {super();
               _createView();
               _createScene(); }
          protected function _createView() : void
               {_camera = new HoverCamera3D();
               _camera.distance = 1000;
               _camera.tiltAngle = 10;
               _camera.panAngle = 180;
               _view = new View3D();
               _view.x = 400;
               _view.y = 300;
               _view.camera = _camera;
               addChild(_view);
               addEventListener(Event.ENTER_FRAME, _onEnterFrame);}
          protected function _createScene() : void
              {// To be overridden}
          protected function _onEnterFrame(ev : Event) : void
              {_camera.panAngle += (stage.mouseX - stage.stageWidth/2) / 100;
               _camera.hover();
               _view.render(); } } }
```

เริ่มต้นด้วยการเพิ่มแพคเกจโดยการใช้ และระบุคลาสใหม่ที่เพิ่มเติมจาก Sprite ในคอนสตรัคเตอร์ มีสองวิธี คือ _createView () ซึ่งตั้งค่า framework พื้นฐาน และ _onEnterFrame () เป็นวิธีการจัดการสำหรับการแสดงผลและ_createScene () ซึ่งเป็นที่สำหรับข้อมูล 3 มิติ จะสร้างขึ้นและเพิ่มเข้าไปในฉาก

4.3.3 การเข้าใจพื้นฐาน

รูปทรงพื้นฐานมีการใช้อย่างแพร่หลาย อาทิ รูปทรงกลมที่อยู่มากมายในเว็บซึ่งเป็นส่วนหนึ่งของงานต่างๆ ระนาบก็สามารถใช้เป็น image cards ในหน้าเวปไซต์ต่างๆ ด้วยวิธีการทางด้านงานสามมิติ วัตถุเหล่านี้สามารถสร้างได้ง่ายโดยการใช้อเว์ยทรีดี

ขั้นตอนแรก พิจารณาที่พื้นฐานโดยสร้างคลาสขึ้นจาก Chapter04SampleBase ดังตัวอย่าง 4.46

ตัวอย่าง 4.46

```
package flash3dbook.ch04
    {import away3d.materials.*;
     import away3d.primitives.*;
     import flash3dbook.ch04.*;
     [SWF(width="800", height="600")]
     public class CommonPrimitives extends Chapter04SampleBase
          { public function CommonPrimitives()}
          { super(); }
     protected override function _createScene() : void
         { // Create default material
         var mat : WireColorMaterial = new WireColorMaterial(0xcccccc); }
     }
```

โค้ดก่อนหน้านี้จะตั้งค่า SWF ด้วยความกว้าง 800 และสูง 600 แทนที่ _createScene() ซึ่งสามารถจะเพิ่มเนื้อหาและสามารถสร้าง WireColorMaterial ได้ใหม่

4.3.3.1 The plane primitive

ระนาบพื้นฐานสามารถสร้างได้จาก 2 มิติ ถึง 3 มิติโดยมีรูปทรงเรขาคณิตซึ่งประกอบด้วยโครงตาข่าย 2 มิติ ซึ่งแสดงในรูปสามเหลี่ยมและสี่เหลี่ยม ด้วยเหตุผลนี้ส่วนใหญ่ระนาบจะใช้ในการเพิ่มรายละเอียดของผลกระทบใน 3 มิติ เพื่อที่จะติดต่อโดยการแสดงเนื้อหา 2

มิติ (ตัวอย่างเช่น รูปภาพหรือข้อความ) ลงบนระนาบพื้นผิว ข้อมูลเหล่านี้สามารถทำการหมุนได้ใน 3 มิติ และใช้งานได้ในการแสดงผล และทำการติดต่อกัน รูปแบบของ interface สามารถอ้างอิงได้เป็นโปสการ์ดในพื้นที่ เพราะว่าระนาบพื้นผิวปรากฏเหมือนโปรการ์ดที่ซึ่งใช้ในการเขียนข้อความ 2 มิติลงไป

พื้นฐานระนาบสามารถสร้างด้วยคลาส Plane จาก away3d.primitives คุณสมบัติของ yUp ระบุไว้ว่าระนาบที่สมมาตรด้วยจุดที่ลากไปตามแกน y หรือแกน z ส่วนใหญ่ พื้นฐานอเว์ยทรีดีจะประกอบด้วยคุณสมบัติ yUp เพื่อที่จะให้มีความแตกต่างกัน ซึ่งปรากฏบน Plane โดยการเพิ่ม _createScene () ดังตัวอย่าง 4.47

ตัวอย่าง 4.47

```
var plane : Plane = new Plane();
plane.yUp = false;
plane.material = mat;
_view.scene.addChild(plane);
```

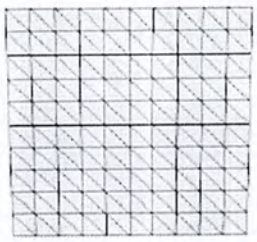
ถึงตอนนี้จะทำการสร้างพื้นฐานระนาบใหม่ โดยยกตัวอย่าง Plane โดยการกำหนดค่า yUp และ material และเพิ่มเข้าไปในฉาก สังเกตุว่าคุณสมบัติของระนาบ material จะถูกตั้งค่าตามค่าที่ระบุไว้ข้างต้น

คอมไพล์ตัวอย่าง CommonPrimitives ที่จุดนี้จะสร้างระนาบด้านหน้าด้วยพื้นผิวสีเทา โครงข่ายระนาบสร้างจากรูปสามเหลี่ยม 2 รูปที่กำหนดค่าไว้แล้ว ในบางสถานการณ์จำเป็นต้องแบ่งย่อยโครงข่ายให้เป็นรูปสามเหลี่ยมเล็กๆ ดังตัวอย่างเช่น ถ้าตั้งใจที่จะเปลี่ยนรูปร่างของระนาบให้เป็นเหมือนกระดาษ นี่คือรูปแบบหนึ่งของการแบ่งย่อย หรือการแบ่งส่วนจะสามารถทำได้โดยการใช้คุณสมบัติ segmentsW และ segmentsH เพื่อที่จะแบ่งย่อยทั้งความหนา และความสูงของระนาบตามลำดับ นำโค้ดของการแบ่งย่อยมาเพิ่มใน _createScene() ดังตัวอย่าง 4.48

ตัวอย่าง 4.48

```
plane.segmentsW = 10;
plane.segmentsH = 10
```

คอมไพล์ CommonPrimitives จะแสดงภาพเหมือนในรูปที่ 4.34



รูป 4.34 ระนาบซึ่งตั้งค่า segmentsW และ segmentsH ไว้ที่ 10

ขนาดของระนาบกำหนดโดยคุณวมบัติของความกว้างและความสูงของ Plane โดยตั้งค่าทั้งสองให้เป็น 100 แต่สามารถปรับเปลี่ยนให้คล้ายคลึงกับสภาวะจริงด้วยคุณสมบัติ segmentsW และ segmentsH โดยการเพิ่มโค้ดดังตัวอย่าง 4.49 ไว้ที่ท้ายของ _createScene()

ตัวอย่าง 4.49

```
plane.width = 200;
plane.height = 200;
```

คอมไพล์ CommonPrimitives จะแสดงภาพเหมือนในรูป 4.34 เพียงแต่ครั้งนี้ความสูงและความกว้างจะเพิ่มขึ้นเป็น 2 เท่า

4.3.3.2 Back-face culling

หยุดการทำงานสักครู่ คอมไพล์โค้ด และหมุนโดยการใช้เมาส์ควมคุมกล้อง เพื่อที่จะให้ระนาบสามารถเห็นได้จากด้านหลัง ที่มุมนี้ระนาบจะไม่สามารถมองเห็นได้ เพราะว่าด้วยเทคนิคนี้ จึงเรียกว่า back-face culling จุดอื่นๆจากกล้องจะไม่ได้รับความสนใจในการแสดงภาพ เนื่องจากกระบวนการนี้ ทิศทางของตำแหน่งด้านหน้า ซึ่งถูกระบุโดยเวคเตอร์ 3 มิติ ซึ่งเรียกว่า normal เวคเตอร์ ซึ่งจะมีอยู่ในอเวย์ทรีดีโดยใช้คุณสมบัติ getter ที่ด้านหน้าของวัตถุ normalเวคเตอร์จะคำนวณเหมือนเวคเตอร์ตั้งฉากบนระนาบของด้านหน้า ทิศทางของจุด normal เวคเตอร์ จะคำนวณ ดังเช่น ถ้าสังเกตุด้านหน้าโดยการมองไปยัง normalเวคเตอร์ (ดังนั้น normalเวคเตอร์จะชี้ออกจากตัว) จุดยอดของด้านหน้าจะถูกเรียงทวนเข็มนาฬิกา ในทางตรงกันข้ามโครงข่ายของวัตถุของรูปสามเหลี่ยมจะถูกวาดไว้บนฉากด้วยการเรียงลำดับจุดยอดตามเข็มนาฬิกา จะถูกยกเว้นจากกระบวรการทำงาน

back-face culling เป็นสิ่งสำคัญ เพราะว่าวัตถุโครงข่ายส่วนใหญ่ (เช่นลูกบาศก์และทรงกลม) จะถูกสร้างทำให้ไม่มีการแบ่งเป็นด้านหน้าหรือด้านหลัง ตัวอย่างเช่น ไม่สามารถ

เห็นภายในกล่อง 3 มิติ ที่สร้างโดยมุมมองของลูกบาศก์นอกจากกล้องจะชี้ไปที่โครงข่ายของลูกบาศก์นั้น

ระนาบเป็นข้อยกเว้นสำหรับกฎข้อนี้ ดังนั้น ควรจะทำการปิด back-face culling สำหรับวัตถุระนาบ สามารถทำวิธีการนี้โดยการตั้งค่าให้เหมาสม ดังตัวอย่าง 4.50

ตัวอย่าง 4.50

```
plane.bothsides = true;
```

เพิ่มโค้ดนี้ใน _createScene() และคอมไพล์ CommonPrimitives อีกครั้ง จะปรากฏทั้งสองด้สนของระนาบ เมื่อมีการควบคุมด้วยเมาส์

4.3.3.3 ลูกบาศก์พื้นฐาน

ลูกบาศก์เป็นอีกรูปทรงหนึ่งที่ใช้กันทั่วไป และถูกสร้างขึ้นในอเว์ยทรีดีโดยคลาส Cube ที่อยู่ในแพคเกจ away3d.primitives ดำเนินการต่อในตัวอย่าง CommonPrimitives เพิ่มโค้ดดังตัวอย่าง 4.51ใน _createScene () สร้างลูกบาศก์ และกำหนดตำแหน่ง 200 หน่วยไปทางขวา

ตัวอย่าง 4.51

```
var cube : Cube = new Cube();
cube.material = mat;
cube.x = 200;
_view.scene.addChild(cube);
```

เพื่อให้เป็นห้องลูกบาศก์ จะทำการเพิ่มโค้ดต่อไปนี้ที่ตำแหน่งระนาบ 200 หน่วยไปทางซ้าย ดังตัวอย่าง 4.52

ตัวอย่าง 4.52

```
plane.x = -200;
```

คอมไพล์ตัวอย่าง CommonPrimitives จะปรากฏลูกบาศก์ และระนาบพื้นฐานในด้านข้าง ลูกบาศก์ใช้คุณสมบัติเช่นเดียวกับระนาบ สำหรับการกำหนดขนาด การแบ่งส่วน และ

สามารถเพิ่มความลึก โดยคุณสมบัติ segmentsD สำหรับกำหนดขนาดความลึกของลูกบาศก์ การเพิ่มโค้ดต่อไปนี้ยังจุดสิ้นสุดของ _createScene () จะแบ่งย่อยวัตถุลูกบาศก์ และเพิ่มขนาดความกว้าง ความสูง และความลึก ดังตัวอย่าง 4.53

ตัวอย่าง 4.53

```
cube.segmentsW = 10;
cube.segmentsH = 10;
cube.segmentsD = 10;
cube.width = 200;
cube.height = 200;
```

4.3.3.4 ทรงกลมพื้นฐาน

ทรงกลมพื้นฐานสร้างขึ้นในอเวย์ทรีดีโดยคลาส Sphere ใน away3d.primitives ขนาดของทรงกลมจะไม่มีความกว้างความสูงใดๆ หรือคุณสมบัติความลึกที่ได้เห็นในรูปทรงก่อนหน้านี้ แต่มีคุณสมบัติเดียวที่เรียกว่ารัศมี ให้เพิ่มโค้ดใน _createScene()ดังตัวอย่าง 4.54

ตัวอย่าง 4.54

```
var sphere : Sphere = new Sphere();
sphere.radius = 50;
sphere.material = mat;
_view.scene.addChild(sphere);
```

คอมไพล์ตัวอย่าง CommonPrimitives จะแสดงทรงกลมที่มีอยู่ในระหว่างลูกบาศก์ที่ปรากฏอยู่แล้วและระนาบของวัตถุ ทรงกลมทางด้านซ้ายในรูปที่ 4.35 จะแสดงผล ตอนนี้จะพบปัญหาด้วยมุมมองของทรงกลม โดยที่ปริมาณของพื้นผิวที่แสดงเล็กน้อย เกิดขึ้นอย่างสมมาตร และทำให้ไม่สามารถมองเห็นได้ การที่จะสร้างรูปทรงกลมให้มีความรายเรียบมากขึ้นจำเป็นต้องใช้จำนวนสามเหลี่ยมมากขึ้นด้วย ซึ่งวิธีการนี้สามารถทำได้ โดยการตั้งค่าการแบ่งส่วนในทางเดียวกับมุมมองข้างต้น โดยใช้คุณสมบัติของ segmentsW และ segmentsH ของมุมมองทรงกลมเพิ่มโค้ดในตัวอย่าง _createScene() ดังตัวอย่าง 4.55

ตัวอย่าง 4.55

```
sphere.segmentsW = 24;
sphere.segmentsH = 12;
```

คอมไพล์ CommonPrimitives จะแสดงรูปทรงกลมที่คล้ายคลึงกับรูปทรงกลมทางด้านขวาดังรูป 4.35 ในโค้ดข้างต้นค่าที่ใช้สำหรับคุณสมบัติ segmentsW และ segmentsH จะไม่เท่ากัน ทั้งที่เป็นลูกบาศก์ และเป็นมุมมองระนาบ เพราะว่าโครงสร้างสำหรับรูปทรงกลมสมมาตรที่มีความแตกต่างกันด้วยระยะทางทั้งหมดระหว่างความสูงเริ่มต้น และความสูงสุดท้ายเป็นครึ่งหนึ่งของระยะทางระหว่างความกว้างเริ่มต้นและความกว้างสุดท้าย ด้วยเหตุนี้การกำหนดค่าที่ดีที่สุดสำหรับการแบ่งส่วนของทรงกลมถูกตั้งไว้ที่ครึ่งหนึ่งของขนาดกว้าง โดยเปรียบเทียบกับขนาดของความสูง ทรงกลม(ซ้าย) ที่ตั้งค่าด้วย segmentsW = 8 segmentsH = 6 ทรงกลม(ขวา) มีระดับของรายละเอียด สูงกว่าด้วย segmentsW = 24 segmentsH = 12

รูปที่ 4.35 เปรียบเทียบการตั้งค่า segmentsW segmentsH ของทรงกลม

4.3.4 พื้นฐานของเส้นและเซกเมนต์

ก่อนหน้านี้ ได้เห็นวิธีการสร้างวัตถุประกอบด้วยองค์ประกอบของ face ในอเวย์ทรีดีแต่ควรที่จะสร้าง wireframe ที่แสดงให้เห็นถึงพื้นฐาน รวมถึงประกอบด้วยโครงสร้างของเส้นที่รวมตัวกันเป็นลูกบาศก์ หรือพื้นฐานต่างๆของระนาบ มีความสามารถที่จะวาดรูปเส้นขึ้นมาได้สำหรับพื้นฐานของ face ดังที่เห็นในตัวอย่างก่อนหน้านี้ โดยใช้ทรงกลมพื้นฐาน แต่นี่ไม่ใช่ในสภาวะอุดมคติ ถ้ามุมของ ภาพต่างๆจากรูปทรงเรขาคณิต มีมากกว่าสามด้าน จึงจำเป็นต้องหาวิธีการใหม่ที่จะสร้างพื้นฐานในอเวย์ทรีดีโดยใช้ส่วนประกอบของส่วนที่เล็กที่สุด วิธีการนี้สามารถทำได้โดยคลาส wire primitive ซึ่งอยู่ใน away3d.primitives

ครั้งแรกสำหรับการสร้าง wire primitives จะสร้างคลาสด็อคคิวเม้นใหม่ขึ้นมา โดยใช้ชื่อมา Chapter04SampleBase ดังตัวอย่าง 4.56

ตัวอย่าง 4.56

```
package flash3dbook.ch04
    {
     import away3d.materials.*;
     import away3d.primitives.*;
     import flash3dbook.ch04.*;
     [SWF(width="800", height="600")]
     public class CommonWirePrimitives extends  Chapter04SampleBase
        {
        public function CommonWirePrimitives()
            {
            super();
            }
         protected override function _createScene() : void
            {// Create default material
             var mat : WireframeMaterial = new
         WireframeMaterial(0x000000);
            } } }
```

ในโค้ดข้างต้น เนื้อหาเริ่มต้นที่ตั้งค่าไว้ที่ WireFrameMaterial โดยสืบเนื่องมาจากคลาส SegmentMaterial และมีความสัมพันธ์กับวัตถุพื้นฐาน

4.3.4.1 Wireframe primitives

โดยส่วนใหญ่มุมมอง Face พื้นฐานจะมีค่าเท่ากันกับมุมมองในพื้นฐานเดียวกันกับ wireframe ในอเวย์ทรีดีมุมมอง wireframe เหล่านี้จะสร้างด้วยชื่อของคลาสที่เหมือนกับมุมมองทั่วไป แต่จะมีคำนำหน้าว่า wire โค้ดดังต่อไปนี้ จะถูกเขียนขึ้นใหม่ เพื่อที่จะเพิ่มเข้าไปใน _createScene() ของตัวอย่าง CommonPrimitives ดังตัวอย่าง 4.57

ตัวอย่าง 4.57

```
var plane : WirePlane = new WirePlane();
plane.yUp = false;
plane.x = -200;
plane.width = 200;
plane.height = 200;
plane.material = mat;
_view.scene.addChild(plane);

var cube : WireCube = new WireCube();
cube.x = 200;
cube.width = 200;
cube.height = 200;
cube.depth = 200;
cube.material = mat;
_view.scene.addChild(cube);

var sphere : WireSphere = new WireSphere();
sphere.radius = 50;
sphere.segmentsW = 24;
sphere.segmentsH = 12;
sphere.material = mat;
_view.scene.addChild(sphere);
```

การใส่โค้ดที่ _createScene() ในตัวอย่าง CommonPrimitives และคอมไพล์ จะแสดงมุมมองสามมุมมองที่เหมือนกัน ดังเช่นที่แล้วมา แต่ในครั้งนี้จะแสดงส่วนที่เหลือโดยใช้ WireframeMaterial

4.3.4.1 Combining wireframe and regular primitives

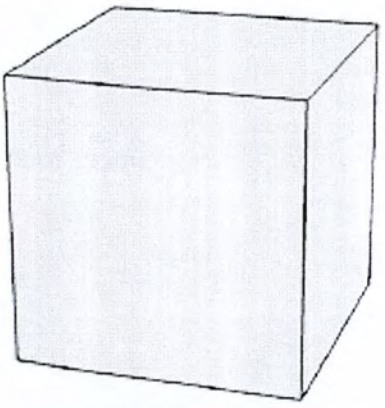
ผลกระทบโดยทั่วๆไป สามารถทำให้สำเร็จด้วยการรวมกันของพื้นฐานด้วย wireframe ของตัวมันเอง ลองวางตำแหน่ง WireCube และ Cube ไว้บนสุดของบรรทัด โดยการสร้างตามตัวอย่างต่อไปนี้ Chapter04SampleBase ดังตัวอย่าง 4.58

ตัวอย่าง 4.58

```
package flash3dbook.ch04
    {import away3d.materials.*;
     import away3d.primitives.*;
     import flash3dbook.ch04.*;
     [SWF(width="800", height="600")]
     public class CombinedWireAndRegularCube extends  Chapter04SampleBase
        {public function CombinedWireAndRegularCube ()
            {super(); }
         protected override function _createScene() : void
            {var wireCube : WireCube = new WireCube();
             wireCube.material = new WireframeMaterial(0x000000);
             _view.scene.addChild(wireCube);
             var regularCube : Cube = new Cube();
             regularCube.material = new ColorMaterial(0xcccccc);
             regularCube.scale(0.99);
             _view.scene.addChild(regularCube); } }
```

สร้าง WireCube และ Cube ที่ตำแหน่งเดียวกันในพื้นที่ WireCube ใช้วิธีการเดียวกันกับ WireframeMaterial ดังตัวอย่างที่เห็นกันมาก่อนแล้ว เพื่อที่จะอธิบายสีของส่วนประกอบใน WireCube ในขณะที่ Cube ทั่วไปใช้ ColorMaterial สำหรับการแสดงสีของ face ที่ใช้ในมุมมอง Cube คอมไพล์ CombinedWireAndRegularCube จะแสดงสิ่งที่ปรากฏใน Cube เดียวด้วยโครงสร้างของมุมสีดำที่แสดงในรูป 4.36 Cube ทั่วไปจะมีขนาด 0.99 ของขนาดปกติ โดยใช้ scale() (ซึ่งสามารถใช้งานได้ในอเวย์ทรีดี) วิธีการนี้กระทำเสร็จ เมื่อต้องการว่าส่วน ประกอบของ WireCube จะซ้อนทับบน face ของ Cube ทั่วๆไป นี่เป็นปริมาณมากพอที่ส่งผลกระทบต่อการคำนวณ Z-sorting แต่ไม่เพียงพอที่จะมองเห็นในฉาก

เปรียบเทียบผลในรูป 4.36 ด้วยตัวอย่างที่แสดงลูกบาศก์ทั่วๆไป ด้วย WireColorMaterial ในตัวอย่าง CommonPrimitives จะเห็นความแตกต่างระหว่างการแสดงผลด้วยพื้นฐาน wire และเค้าโครงของ face ด้วยพื้นฐานทั่วไป



รูป 4.36 ลูกบาศก์ที่สร้างโดยการใช้งานของ wire

4.3.4.3 การสร้างเส้นที่ไม่มีพื้นฐานแน่นอน

ส่วนประกอบของสิ่งที่เล็กๆ สามารถพัฒนาโครงสร้างของรูปมาตรที่กำหนดไว้แล้ว โดยใช้คลาส LineSegment ซึ่งปรากฏอยู่ใน away3d.primitives โดยการสร้างเส้น 3 มิติ ในพื้นที่ โดยระบุจุดของเวคเตอร์เริ่มต้นและเวคเตอร์สุดท้ายไว้อย่างตายตัว ในการทดลองนี้ จะสร้างคลาส ด็อคคิวเม้น โดยเพิ่มมาจาก Chapter04SampleBase ดังตัวอย่าง 4.59

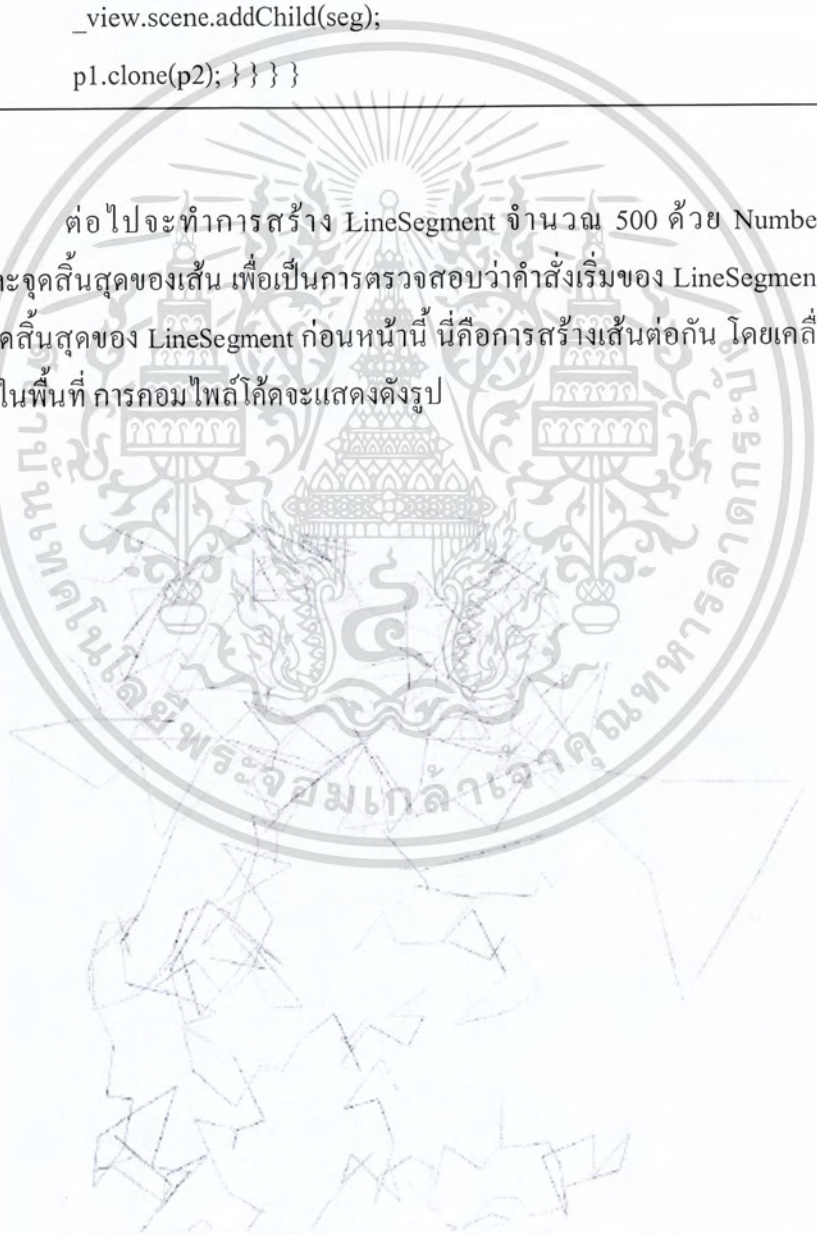
ตัวอย่าง 4.59

```
package flash3dbook.ch04
    {import away3d.core.math.*;
     import away3d.primitives.*;
     import flash3dbook.ch04.*;
     [SWF(width="800", height="600")]
     public class LinesInSpaceWithLineSegment extends    Chapter04SampleBase
        {public function LinesInSpaceWithLineSegment ()
            { super(); }
         protected override function _createScene() : void
            {var i : int, p1 : Number3D, p2 : Number3D, seg : LineSegment;
             p1 = new Number3D();
             p2 = new Number3D();
             for (i=0; i < 500; i++)
```

ตัวอย่าง 4.59 (ต่อ)

```
{p2.x = (Math.random()-0.5) * 200;
 p2.y = (Math.random()-0.5) * 200;
 p2.z = (Math.random()-0.5) * 200;
 p2.add(p2, p1);
 seg = new LineSegment();
 seg.start = p1;
 seg.end = p2;
 _view.scene.addChild(seg);
 p1.clone(p2); } } } }
```

ต่อไปจะทำการสร้าง LineSegment จำนวณ 500 ด้วย Number3D สำหรับจุดเริ่มต้นและจุดสิ้นสุดของเส้น เพื่อเป็นการตรวจสอบว่าคำสั่งเริ่มของ LineSegment จะทำหน้าที่เหมือนกับจุดสิ้นสุดของ LineSegment ก่อนหน้านี้ นี่คือการสร้างเส้นต่อกัน โดยเคลื่อนที่ระหว่างจุดแรนดอมในพื้นที่ การคอมไพล์โค้ดจะแสดงดังรูป



รูป 4.37 การสร้างเส้นแบบสุ่มจากตัวอย่าง LinesInSpaceWithLineSegment

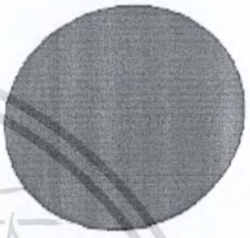
4.3.5 Using regular polygons

ที่ผ่านมาพูดถึงมุมมองของระนาบที่เป็นรูปทรงสี่เหลี่ยม และสี่เหลี่ยมมุมฉาก โดยขีดจำกัดของคลาส Plane ในอเวย์ทรีดี แต่ยังสามารถสร้างระนาบด้วยด้านไม่จำกัด โดยใช้ RegularPolygon ผลลัพธ์ที่ได้ของรูปทรงอื่นๆ จะหมุนอย่างเป็นสมมาตรรวมโครงข่ายให้เป็นรูปร่างของรูปห้าเหลี่ยม หกเหลี่ยม แปดเหลี่ยม หรือรูปทรงอื่น จำนวณของด้วนจะถูกกำหนดด้วยขนาดของ RegularPolygon ตัวอย่างของโค้ดที่เพิ่มจาก Chapter04SampleBase ดังตัวอย่าง 4.60

ตัวอย่าง 4.60

```
package flash3dbook.ch04
    {import away3d.materials.*;
     import away3d.primitives.*;
     import flash3dbook.ch04.*;
     [SWF(width="800", height="600")]
     public class PolygonsWithRegularPolygon extends      Chapter04SampleBase
        {public function PolygonsWithRegularPolygon ()
           {super();}
         protected override function _createScene() : void
           {//create a pentagon
           _createPoly(5, 0xdddddd, -250);
           //create a dodecahedron
           _createPoly(12, 0x999999, 0);
           //create a circle
           _createPoly(100, 0x222222, 250); }
         protected function _createPoly(sides : int, color : int, x : Number) : void
           {var polygon : RegularPolygon;
            polygon = new RegularPolygon();
            polygon.sides = sides;
            polygon.material = new ColorMaterial(color);
            polygon.x = x;
            polygon.yUp = false;
            polygon.bothsides = true;
            _view.scene.addChild(polygon); } } }
```

ในโค้ดข้างต้น การสร้าง RegularPolygon ได้แบ่งแยกไปตามคลาสต่างๆ ที่เรียกว่า _createPoly() ซึ่งต้องเป็นไปตามข้อกำหนดทั้งสามคือ ด้านข้าง สี และ X แต่ละRegularPolygon จะสร้างภายในวิธีการนี้โดยใช้เงื่อนไข sides สำหรับจำนวณทั้งหมดของด้านข้างวัตถุ และ color สำหรับสีของวัตถุ และ X คือตำแหน่งในแกน X นอกเหนือจากคุณสมบัติเหล่านี้ แต่ละวัตถุใหม่ จะมี yUp และตั้งค่า false และ bothsides ในทางเดียวกันกับ Plane ที่อยู่ในตัวอย่าง CommonPrimitives ใน _createPoly() กับ RegularPolygon ทั้งสามจะถูกสร้างขึ้นโดยด้านจำนวณมากมาย และสีต่างๆ การคอมไพล์จะแสดงดังรูป 4.38



รูป 4.38 การสร้างรูปทรงต่างๆ ที่ใช้คลาส RegularPolygon class

เช่นเดียวกับที่พื้นฐานในก่อนหน้านี้ RegularPolygon จะมีสัดส่วนของเส้นที่มีความเท่ากับเรียกว่า WireRegularPolygon หากต้องการดูการทำงาน ให้แทนที่ _createPoly () ด้วยโค้ดดังตัวอย่าง 4.61

ตัวอย่าง 4.61

```
var wirePolygon : WireRegularPolygon;


wirePolygon = new WireRegularPolygon();
wirePolygon.sides = sides;
wirePolygon.material = new WireframeMaterial(color);
wirePolygon.x = x;
wirePolygon.yUp = false;
wirePolygon.bothsides = true;


view.scene.addChild(wirePolygon);
```

ต่อไปจะแทนที่ RegularPolygon ด้วย WireRegularPolygon และแทนที่ ColorMaterial ด้วย WireframeMaterial ผลลัพธ์คือรูปทรงเรขาคณิตที่คอมไพล์ โดยขั้นพื้นฐานจะวาดโดยใช้ส่วนประกอบชิ้นๆเล็ก แทนการใช้ face เพื่อจะเป็นโครงส่วนประกอบของโครงสร้าง

บทที่ 5
แอพพลิเคชั่น

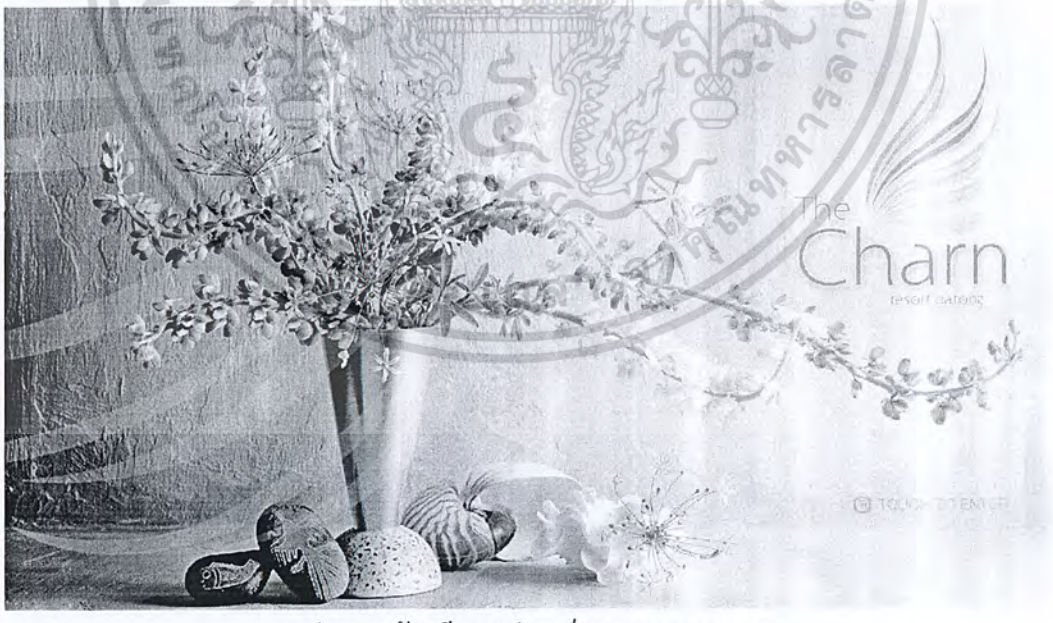
5.1 แอพพลิเคชั่นการนำเสนอแบบอาคาร และห้อง

แอพพลิเคชั่นที่ได้ศึกษา และนำมาเขียนโปรแกรม โดยโปรแกรมแฟลช และเฟล็กส์บิวเดอร์เพื่อเพิ่มความน่าสนใจ จึงต้องมีการเพิ่มการเคลื่อนไหวของภาพ และการจัดวางองค์ประกอบทั้งหมด รวมถึงการออกแบบ ให้มีความน่าดึงดูด ตลอดจนอนิเมชันต่างๆ ซึ่งล้วน แล้วแต่มี ภาษาแอกชันสคิปท์เข้ามามีบทบาททั้งสิ้น

งานออกแบบนี้เน้นรูปแบบที่เรียบง่าย สวยงาม และออกแบบเพื่อให้ใช้งานง่าย สำหรับบุคคลทุกประเภท โดยไม่ต้องศึกษาก่อนใช้งาน และออกแบบมาให้ใช้สำหรับจอสัมผัสด้วย ซึ่งผลงานแบ่งออกเป็น 6 ส่วน ได้แก่

5.1.1 Intro และ Screensaver

เป็นส่วนของหน้าแรกที่เป็นภาพ Screensaver และมีตัวอย่างภาพของห้องต่างๆ มาแสดงโดย อัตโนมัติ ซึ่งจะมีภาพหมุนเวียนสับเปลี่ยนขึ้นมาแสดง เมื่อคลิกเมาส์จะเข้าสู้หน้าหลักของแอพพลิเคชั่น



รูป 5.1 หน้าสกีนเซฟเวอร์(Screen Server)

เมื่อแอพพลิเคชั่นไม่ได้รับการตอบสนองเป็นเวลาประมาณ 60 วินาที แอพพลิเคชั่นจะกลับมาสู้หน้าสกีนเซฟเวอร์นี้ โดยอัติโนมัติ และทำการแสดงภาพในแกลลอรีออกเป็นชุดๆ ชุดละ 3

ภาพ ซึ่งมีการจัดหมวดหมู่ของภาพ และจำแนกภาพจาก XML โดยโค้ดด้านล่างนี้เป็น XML การจัดการภาพหน้าสกีนเซฟเวอร์ เป็นดังตัวอย่าง 5.1

ตัวอย่าง 5.1

```
<?xml version="1.0" encoding="utf-8"?> <feed>
        <project>
                <title>Resort Sea View !</title>
                <image>intro/resort1.jpg</image>
                <image>intro/resort3.jpg</image>
                <image>intro/resort4.jpg</image>
        </project>
        <project>
                <title>Swimming Pool</title>
                <image>intro/resort2.jpg</image>
                <image>intro/resort8.jpg</image>
                <image>intro/resort9.jpg</image>
        </project>
        <project>
                <title>Exterior Design</title>
                <image>intro/resort5.jpg</image>
                <image>intro/resort6.jpg</image>
                <image>intro/resort7.jpg</image>
        </project>
        <project>
                <title>Room Style</title>
                <image>intro/room1.jpg</image>
                <image>intro/room2.jpg</image>
                <image>intro/room3.jpg</image>
        </project>
        <project>
                <title>Room Style</title>
                <image>intro/room4.jpg</image>
                <image>intro/room5.jpg</image>
                <image>intro/room6.jpg</image>
        </project>
</feed>
```

จากการเขียนโค้ดข้างต้น จะทำให้แอพพลิเคชั่นมีการนำภาพมาแสดง โดยการหมุนวนที่ละ 3 ภาพ ดังที่เห็นในรูป

รูป 5.2 สกีนเซฟเวอร์เมื่อมีภาพประกอบ

5.1.2 โฮมเพจ(Home page)

เมื่อทำการคลิกจะเข้าสู้หน้าหลัก ซึ่งเป็นหน้าที่แสดงภาพรวมของโครงการ ทัศนคติ อธิบายโครงการแบบคร่าวๆ และจุดประสงค์ของโครงการ โดยมีการ transition ภาพ equalizer โดยอาศัยเทคนิคของภาษาแอกชันสคิปท์หมุนเวียนเป็นจำนวน 4 ภาพ

รูป 5.3 หน้าโฮม(Home)

5.1.3 แกลลอรี่(Gallery)

แสดงภาพของบรรยากาศรอบๆ

รูป 5.4 หน้าแกลลอรี่(Gallery)

ในส่วนของหน้าแกลลอรีนั้น แบ่งออกเป็น สามหมวดหมู่แกลลอรีเพื่อแสดงแต่ละส่วนของอาคารได้แก่

5.1.3.1 Exterior design

รวบรูปภาพจากภายนอกอาคาร การออกแบบภายนอกอาคารทั้งหมด ทุกมุมไม่ว่าจะเป็นส่วนของดาดฟ้าอาคารที่มีสระว่ายน้ำ , ลอบบี้ของอาคาร หรือห้องพักส่วนตัวไพรเวท (private) รวมถึงส่วนของห้องรับแขกอาคาร และ บาร์

5.1.3.2 Intertior design

ภาพการออกแบบภายในห้องพัก แต่ละแบบ รวมถึงเฟอร์นิเจอร์และสิ่งอำนวยความสะดวกอย่างครบครัน

5.1.3.3 Beach

ภาพบรรยากาศรอบๆ รีสอร์ท หาดทรายของภูเก็ต และ บรรยากาศยามเย็น ซึ่งจัดวางอย่างเหมาะสมและลงตัวภายในแอพพลิเคชั่น

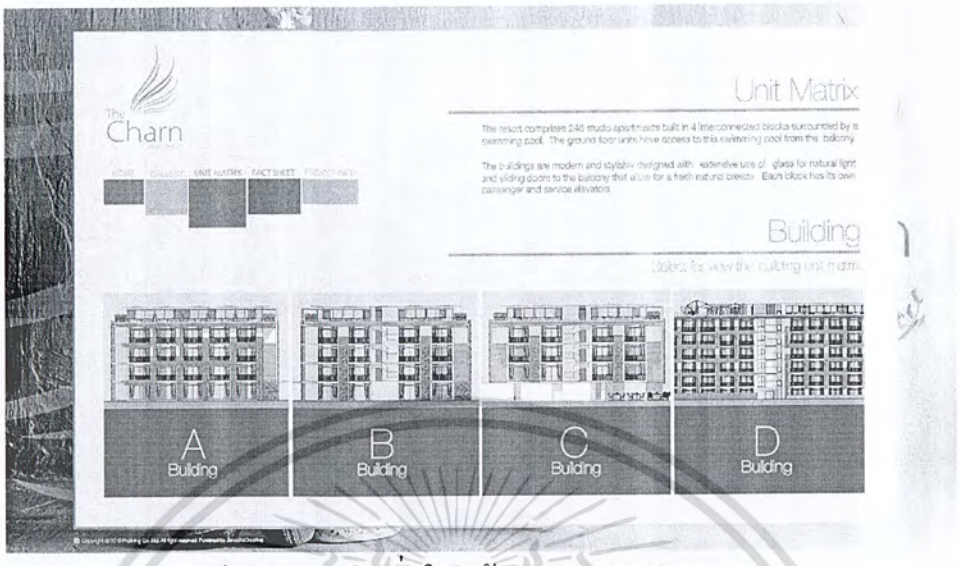
รูป 5.5 หน้าของแอพพลิเนที่แสดงบรรยากาศโดยรวม

รูป 5.6 หน้าของแอพพลิเนที่แสดงบรรยากาศภายในห้อง

เมื่อคลิกแต่ละหมวดหมู่ก็จะเป็นการเปิดหน้าต่างใหม่ของแกลลอรีซึ่งแสดงแบ่งแยกหมวดหมู่ และมีเนื้อหาอธิบายภาพของแต่ละภาพ รวมถึงฟังก์ชั่นในการเลือกและสับเปลี่ยนภาพไปเรื่อยๆ ภายในแต่ละหมวดและสามารถเปลี่ยนหมวดหมู่เพื่อรับชมภาพได้เลย จาก Nav Bar ด้านซ้าย ซึ่งการทำงานของระบบแกลลอรีใช้การดึงข้อมูลภาพจาก XML เช่นเดียวกับ Intro

เมื่อต้องการเพิ่มภาพ และแก้ไขภาพก็สามารถทำได้ผ่าน XML โดยไม่ต้องอาศัยการแก้ Source code ในโปรแกรมเลย

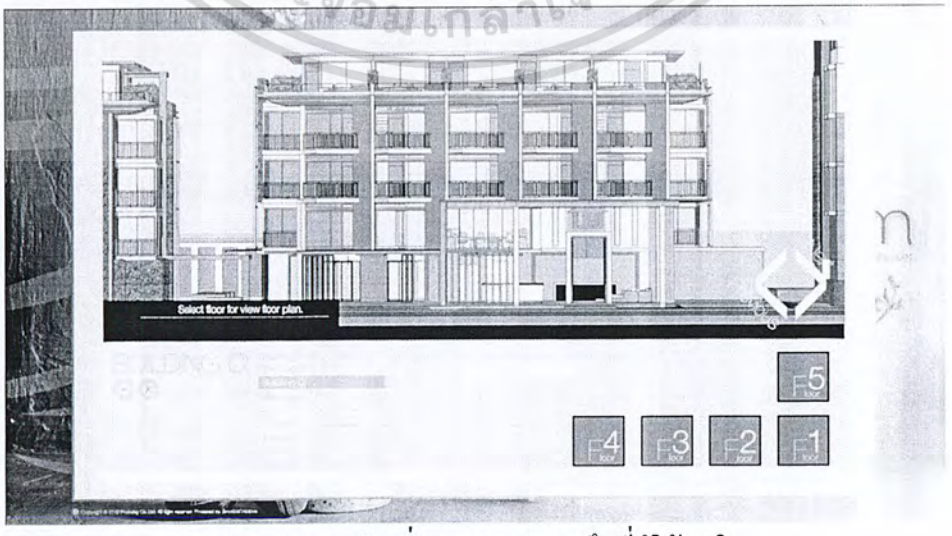
5.1.4 Unit Matrix



รูป 5.7 แอพพลิเคชั่นในหน้าของการนำเสนออาคาร

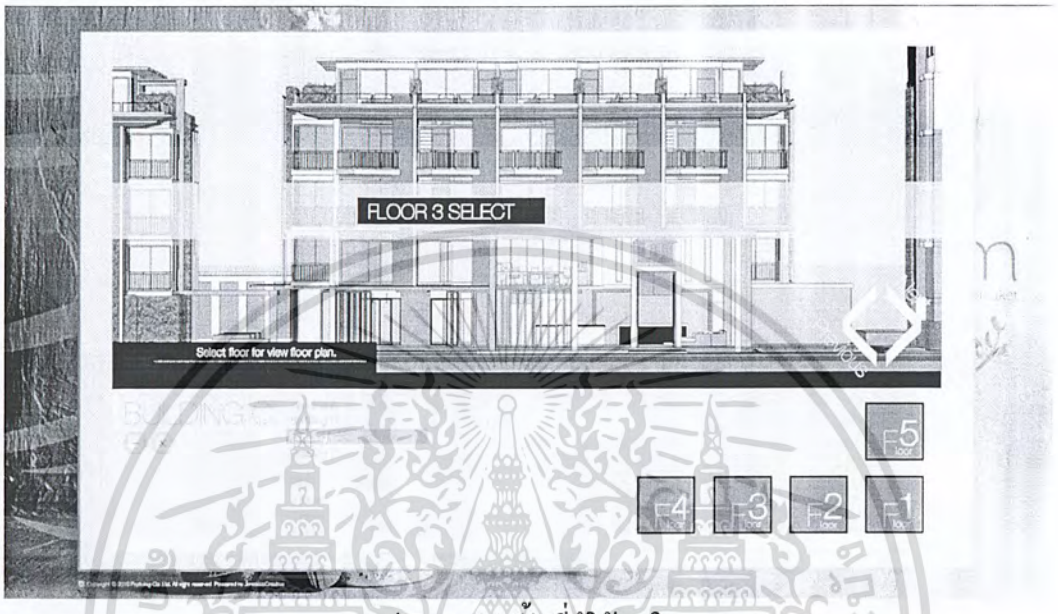
ในส่วนของหน้า Unit Matrix เป็นหัวใจสำคัญของ Application ชิ้นนี้ เนื่องการการนำเสนอต้องการเน้นแสดงอาคารให้ชัดเจนและสามารถให้ผู้ใช้งานได้เห็นภาพจริง และ ตัดสินใจได้ในส่วนของ Interactive นี้จะมีตึกให้เลือกอยู่ 4 ตึกในโครงการได้แก่

1) A Building
2) B Building
3) C Building
4) D Building



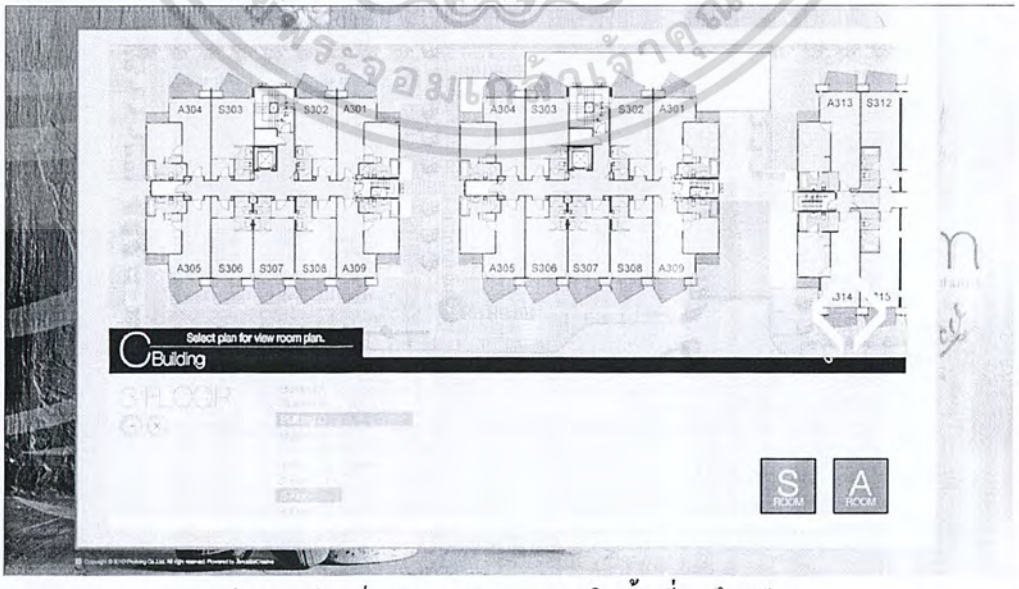
รูป 5.8 แอพพลิเคชั่นจะแสดงภาพตึกที่ผู้ใช้สนใจ

เมื่อเลือกตึกที่สนใจ ก็จะทำการเปิดหน้าต่าง PopUp สำหรับการดูรายละเอียดของตึกนั้น และแสดงจำนวนชั้นของตึกนั้น เช่น ตึก ซี ชั้น 1,2 เป็น Lobby ด้านหน้า และมี 5 ชั้น ผู้ใช้งานสามารถเลือกชั้นเพื่อเข้าไปชมฟลอร์แพลนได้เลยทั้งจากตัวตึก และ จากปุ่มด้านล่าง รวมถึงสามารถเปลี่ยนตึกที่เลือกตั้งแต่ต้นไปเป็นตึกอื่นๆได้ตัวจาก Nav Bar ด้านซ้าย



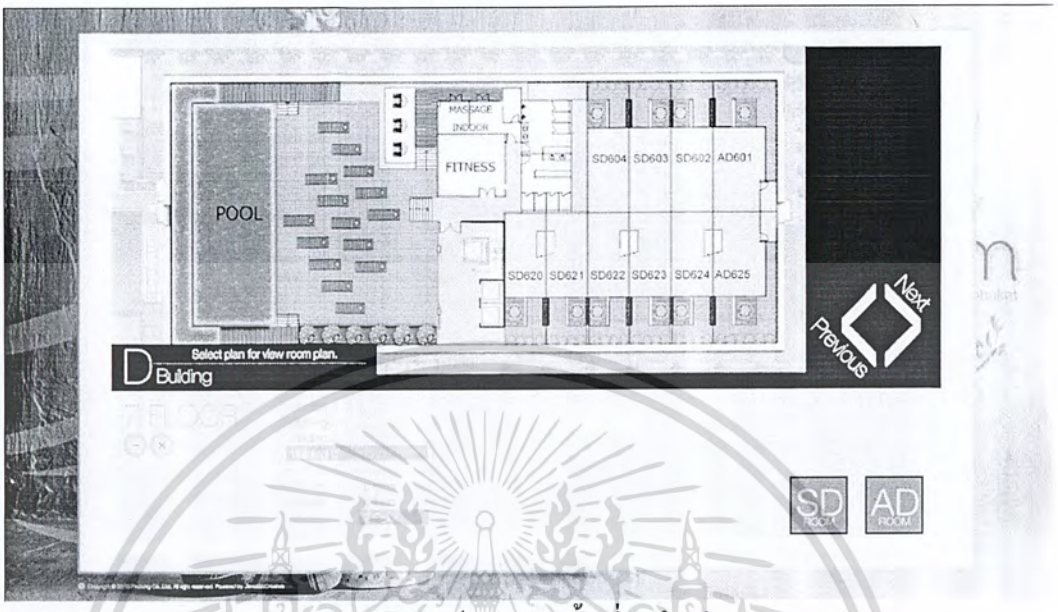
รูป 5.9 แสดงชั้นที่ผู้ใช้สนใจ

เมื่อผู้ใช้งานทำการเลือกชั้น จากตึก ก็จะมีชั้นกำกับเพื่อบอกว่า ผู้ใช้งานเลือกชั้น 3 และทำการ fade เปลี่ยนรูปแบบไปแสดงฟลอร์แพลนสำหรับชั้น 3 ของตึก ซี



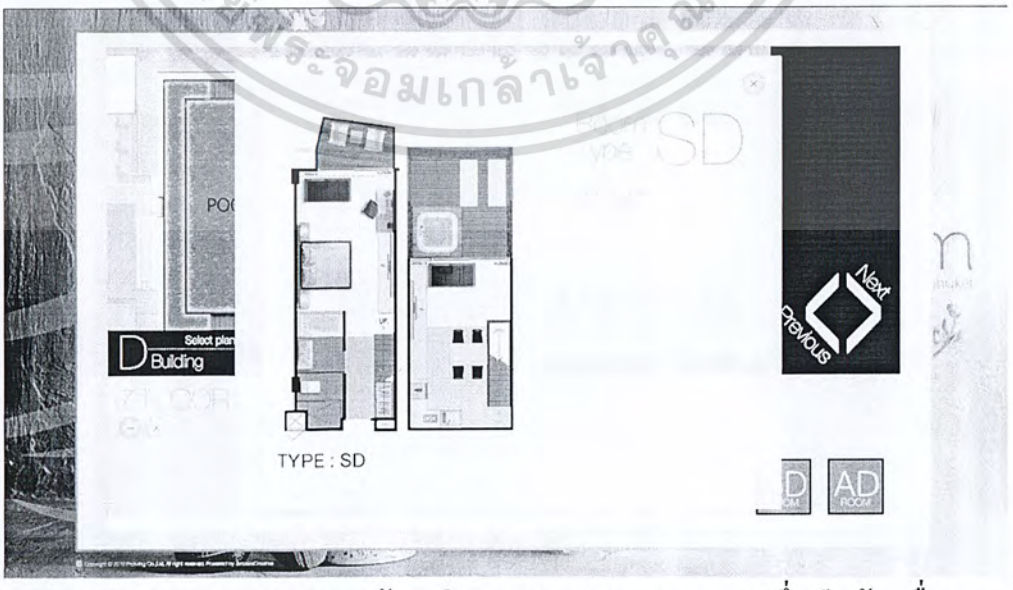
รูป 5.10 ฟลอร์แพลน(Floor Plan) ในชั้นที่ 3 ตึก ซี(C)

ฟลอร์แพลนซึ่งแสดงรายละเอียด และตำแหน่งห้อง รวมถึงตำแหน่งสำคัญต่างๆของชั้นและ ตึกนั้นๆ



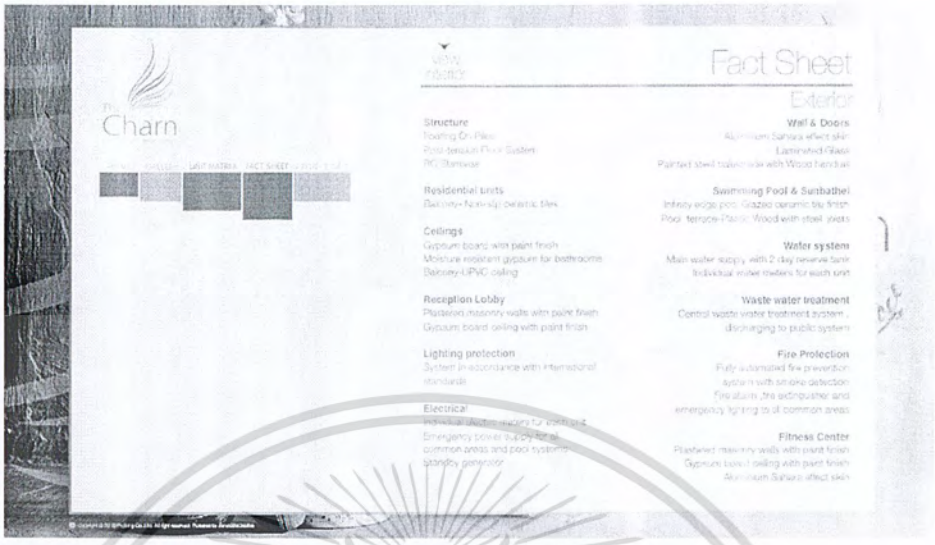
รูป 5.11 ฟลอร์แพลนในชั้นที่ 7 ตึก ดี(D)

ตัวอย่างฟลอร์แพลนของตึก ดี(D) ซึ่งแสดงที่ชั้น 7 ซึ่งเป็นดาดฟ้า และมีสระว่ายน้ำ ห้องฟิตเนส(Fitness) ละที่นอนตากแดดพักผ่อน รวมถึงห้องส่วนตัวแบบไพรเวทจาคุซซี่(Jacuzzi) ซึ่งเป็นห้องเชื่อมต่อระหว่างชั้น 6-7 เมื่อเราทำการคลิกที่ห้องนั้น ก็จะแสดงรูมแพลนที่แสดงการเชื่อมต่อออกมาด้วย



รูป 5.12 รูมแพลน(Room Plan) ของห้องชนิด SD ขนาด 92 ตารางเมตร ซึ่งเป็นห้องเชื่อมสองชั้น

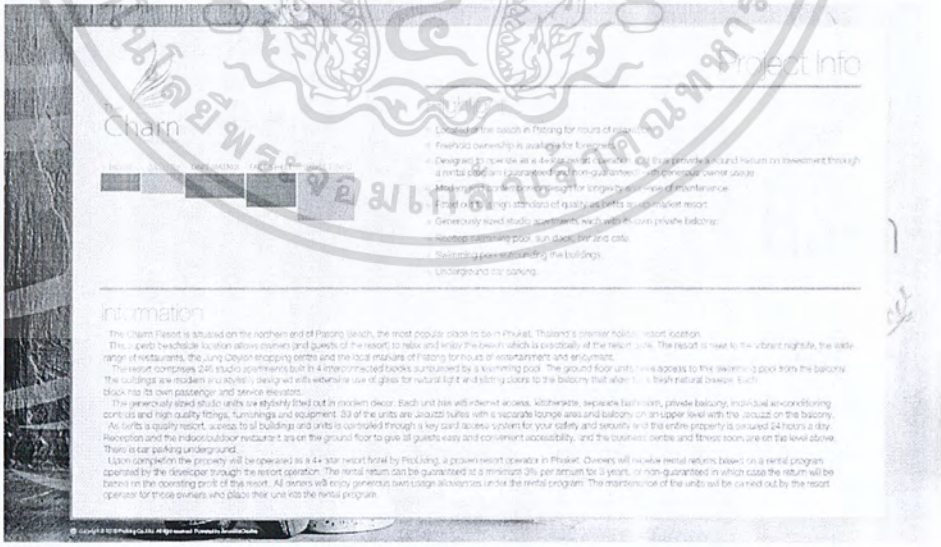
5.1.5 แฟกชีท(Fact Sheet)



รูป 5.13 Fact Sheet

ส่วนของแฟกชีทเป็นส่วนของการอธิบายโครงสร้างทั้ง Exterior ซึ่งแสดงถึงสิ่งอำนวยความสะดวกภายนอกอาคาร รวมถึงวัสดุที่ใช้ และอื่นๆ และ Interior แสดงถึงสิ่งอำนวยความสะดวกภายในห้องพัก วัสดุอุปกรณ์ต่างๆ

5.1.6โปรเจกอินโฟ(Project Info)



รูป 5.14 โปรเจกอินโฟ(Project Info)

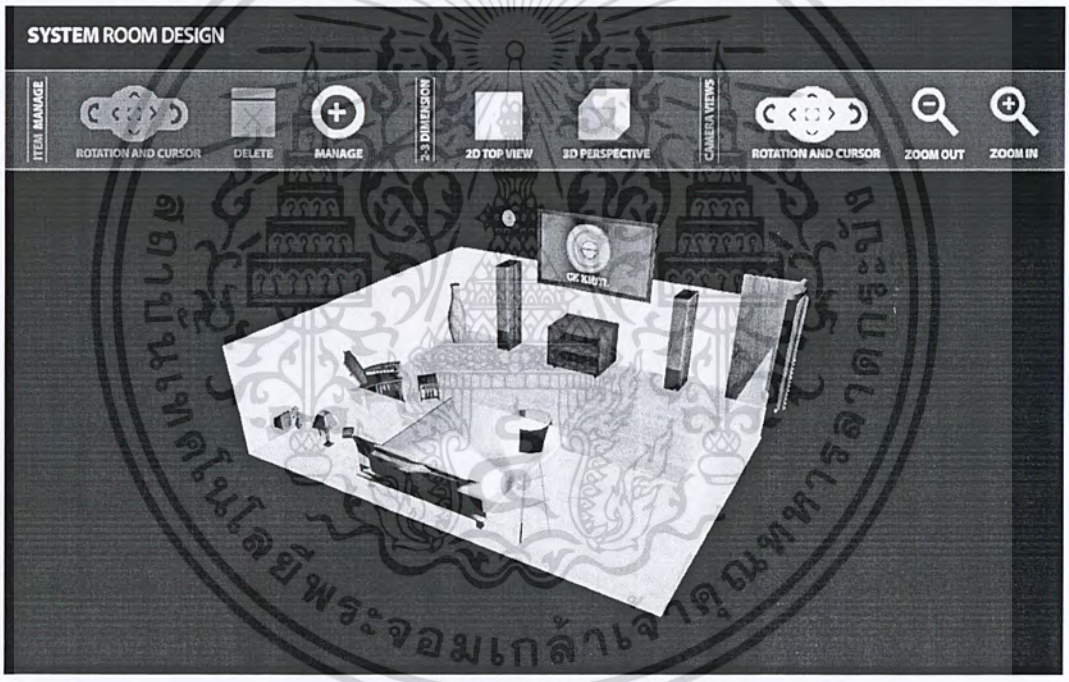
ส่วนของโปรเจกอินโฟเป็นการอธิบายรายละเอียดของโครงการทั้งหมด รวมถึงการจัดทำโครงการ และจุดประสงค์ต่างๆ รวมถึงการแสดง Highlight ของโครงการ

5.2 แอพพลิเคชั่นของห้อง 3 มิติ

แอพพลิเคชั่นภายในห้อง 3 มิติผู้ใช้สามารถออกแบบภายในได้ตัวตัวเอง โดยการเลือกเฟอร์นิเจอร์ ปรับเปลี่ยนตำแหน่ง หรือทำการหมุนเฟอร์นิเจอร์ ให้เป็นไปตามที่ต้องการได้ สามารถเปลี่ยนลายพื้นหรือผนังห้องได้ตามต้องการ โดยตัวโปรแกรมออกแบบมาให้ใช้งานง่าย และมีการใช้คีย์บอร์ดซึ่งเป็นคีย์ลัดในการใช้งาน เพื่อความสะดวกมากขึ้น

จากการที่ผู้ใช้สามารถออกแบบห้องเอง ก่อนที่จะตกแต่งห้องจริง ทำให้ช่วยในการลดค่าใช้จ่าย และผู้ใช้ก็จะได้รับความพึงพอใจ เพราะได้เห็นการจำลองห้องก่อน ที่จะตัดสินใจลงมิตกแต่งจริง

เมื่อเปิดโปรแกรมจะแสดงหน้าหลักของการทำการดังเช่นในรูปนี้

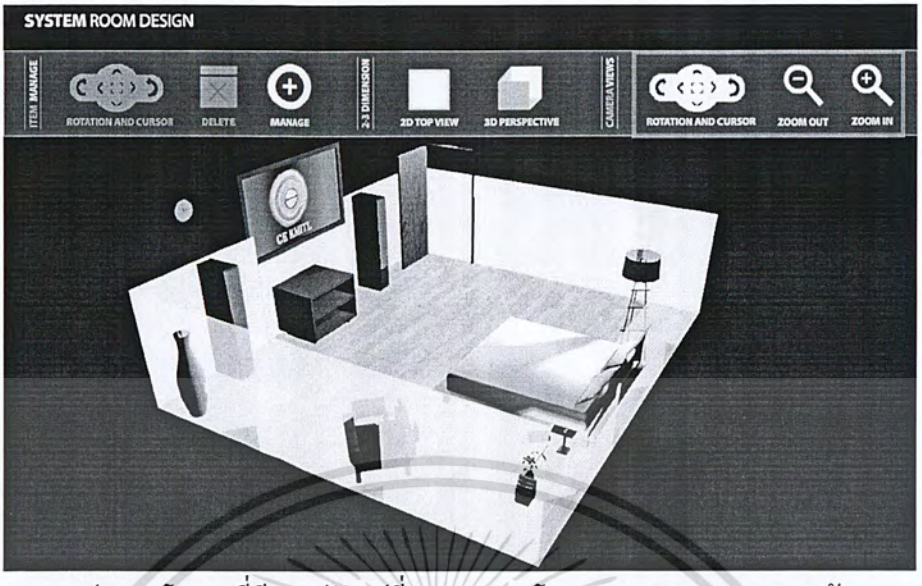


รูป 5.15 หน้าหลักของการตกแต่งห้อง 3D

จากที่กล่าวมาแล้ว ในตัวของแอพพลิเคชั่น สามารถทำงานต่างๆได้หลากหลาย โดยจะมีการทำงานของเครื่องมือ และการใช้งานของแอพพลิชั่น

โดยจะแบ่งเป็นหัวข้อต่างๆ ดังต่อไปนี้

1) ซูม (zoom)และการปรับเปลี่ยนมุมมอง แอพพลิเคชั่นสามารถซูมเข้าและออกได้ตามต้องการ ปรับเปลี่ยนมุมมอง ได้ 360 องศา โดยใช้แถบเครื่องมือที่อยู่ด้านบน หรือคีย์ลัดลูกศรในการหมุนภาพ และการซูมเข้าโดยใช้ Z กับซูมออกโดยใช้ X



รูป 5.16 โมเดลที่มีการปรับเปลี่ยนมุมมอง โดยการหมุน และ zoom เข้า

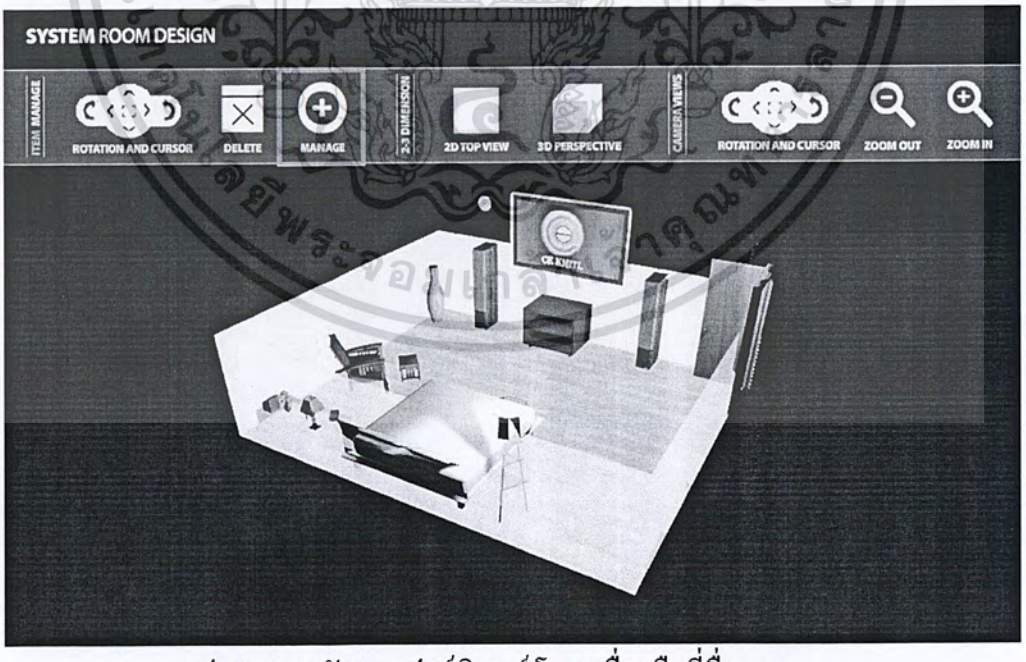
2) มุมมองอัตโนมัติคือด้าน Top และ Perspective ในตัวแอพพลิเคชั่น มีการตั้งค่ามุมมองด้านบน และ แบบ Perspective โดยให้เลือกเป็นมุมมองพื้นฐาน อยู่ในแถบเครื่องมือของแอพพลิเคชั่น

รูป 5.17 มุมมองด้านบนที่ตั้งไว้เป็นมุมมองพื้นฐาน

3) การหมุนเฟอร์นิเจอร์ และลบเฟอร์นิเจอร์ สามารถเลือกเฟอร์นิเจอร์ชิ้นที่ต้องการหมุน หรือลบออกได้ โดยจะปรากฏกรอบสีขาวขึ้นที่เฟอร์นิเจอร์ที่ผู้ใช้เลือก

รูป 5.18 การหมุนเฟอร์นิเจอร์ และลบเฟอร์นิเจอร์ โดยเลือกที่ทีวี

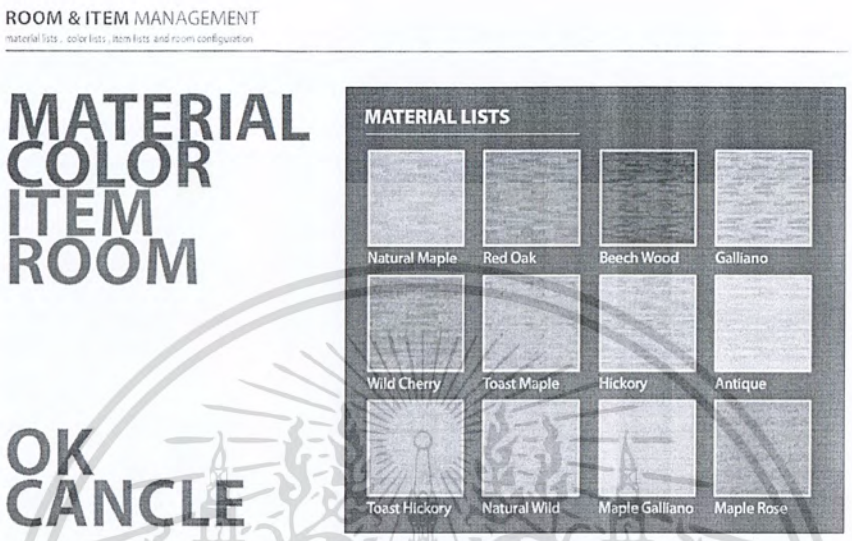
4) การจัดการเฟอร์นิเจอร์ และพื้นผิวต่างๆ โดย manage ในแอพพลิเคชั่นมีเครื่องมือที่ชื่อว่า manage ไว้ในการเปลี่ยนเฟอร์นิเจอร์ หรือเพิ่มเฟอร์นิเจอร์เข้ามา อีกทั้งยังสามารถเปลี่ยนสีผนังในห้อง และเปลี่ยนพื้นของห้องได้



รูป 5.19 การจัดการเฟอร์นิเจอร์ โดยเครื่องมือที่ชื่อ manage

หน้าต่างย่อยของ manage

1) Material สามารถเลือกเปลี่ยนพื้นห้องโดยเลือกคำสั่ง Material จะมีภาพของพื้นห้องมาให้ผู้ใช้เลือก

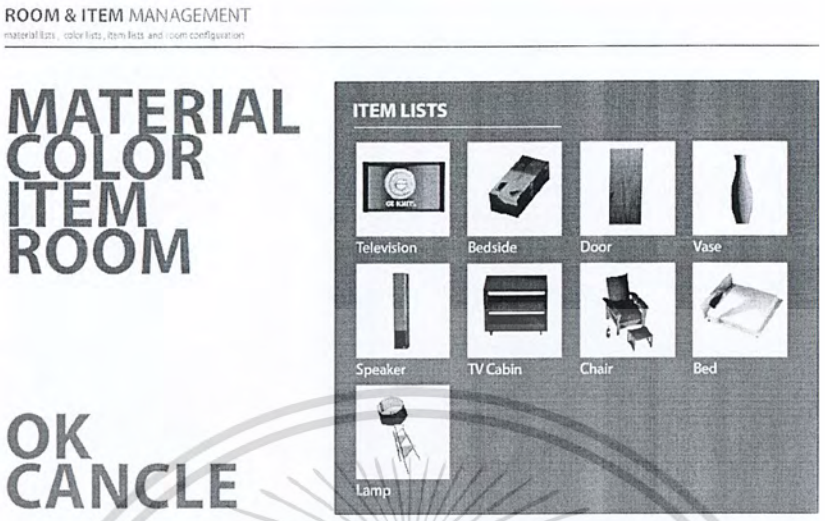


รูป 5.20 หน้าต่างในการเลือก Material

2) Color สามารถเลือกเปลี่ยนผนังหนัง โดยเลือกคำสั่ง Color จะมีสีมาให้ผู้ใช้เลือกได้ตามต้องการ

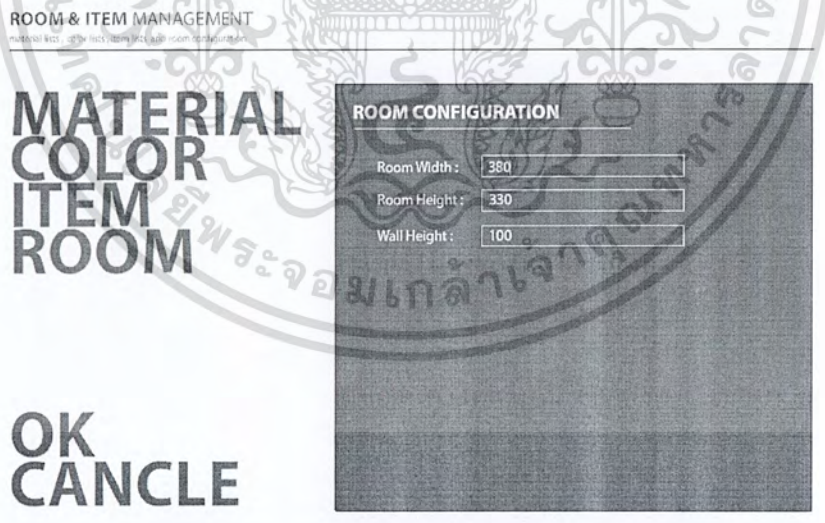
รูป 5.21 หน้าต่างในการเลือกสี

3) Item จะเป็นในส่วนของการเพิ่มเฟอร์นิเจอร์เข้าไปในแอพพลิเคชั่น



รูป 5.22 หน้าต่างที่ใช้ในการเพิ่มเฟอร์นิเจอร์

4) รูม(Room) สามารถเลือกขนาดของห้องได้ตามที่ผู้ใช้ต้องการ โดยการตั้งค่าความกว้าง ความยาว และความสูงของห้องเป็น pixel



รูป 5.23 หน้าต่างในการตั้งค่าขนาดของห้อง

แอพพลิเคชั่นสามารถใช้การนำเสนอผลงานได้อย่างมีประสิทธิภาพ และใช้ในงานออกแบบห้อง 3 มิติ ได้ตามความต้องการของผู้ใช้

บทที่ 6
สรุป

6.1 สรุป

การนำแฟลช และเฟล็กส์ เข้ามาใช้ในงานของการเขียนแอพพลิเคชั่น จำเป็นที่จะต้องศึกษาฟังก์ชั่นต่างๆ และมีความรู้ในระดับหนึ่งแฟลชเป็นโปรแกรมที่สามารถสร้างผลงานให้มีรูปแบบที่สวยงาม สามารถใส่ภาพเคลื่อนไหว และเสียงลงไปในแอพพลิเคชั่นได้

อเวย์ทรีดีซึ่งใช้สำหรับการสร้างผลงานสามมิติ สามารถรองรับโปรแกรมที่ใช้จำลองภาพสามมิติได้มากมาย แต่ที่เลือกใช้คือโปรแกรมทรีดีเอสแมกซ์ซึ่งสามารถสร้างผลงานออกมาได้สมจริง และการนำไฟล์มาใช้ในอเวย์ทรีดีได้ใช้ไฟล์ .3ds ซึ่งแปลงมาจากโปรแกรมทรีดีเอสแมกซ์ซึ่งเป็นไฟล์มาตรฐานของงานสามมิติทั่วไป การเขียนโค้ดอเวย์ทรีดีนั้น ต้องอาศัยความรู้ และความเข้าใจเพื่อสร้างแอพพลิเคชั่นนี้ขึ้น

ตัวแอพพลิเคชั่นที่สร้างขึ้นมามีรูปแบบที่สวยงาม ใช้งานง่าย และสามารถสื่อความเป็นแบนด์ของบริษัทนั้นๆ ได้เป็นอย่างดี สามารถดูรายละเอียดของของตึก ห้อง และรูปแบบของการตกแต่งห้องได้ มีการจำลองห้องเป็น 3 มิติ เพื่อให้ผู้ที่ต้องการออกแบบห้องของตัวเอง สามารถลองออกแบบในห้องจำลอง 3 มิติก่อนได้ โดยมีเฟอร์นิเจอร์ให้เลือก สามารถหมุน และเคลื่อนย้ายตำแหน่ง และดูห้องในมุมมองๆ ได้

6.2 ปัญหาและอุปสรรค

1) การจะโหลดไฟล์เข้าไปในแฟลชเพื่อเสนอในรูปแบบของแอพพลิเคชั่น เป็นเรื่องที่มีปัญหาอย่างมาก เนื่องจากไฟล์ภาพสามมิติมีขนาดใหญ่เกินไป และมี polygon ค่อนข้างจะละเอียด ทำให้ภาพกระตุก หรือไม่สามารถโหลดได้
2) การนำไฟล์ 3 มิติ มาใช้ติดปัญหาในเรื่องของพื้นผิววัตถุ ซึ่งจะทำการโหลดไฟล์มาใช้ในแฟลชตรงๆ เลยไม่ได้
3) สเป็คคอมพิวเตอร์มีที่ไม่อำนวยต่อการทำงานสามมิติมากนัก เพราะการใช้โปรแกรมสามมิตินั้น คอมพิวเตอร์จะต้องมีสเป็คเครื่องที่มากพอ

6.3 แนวทางการแก้ไข

1) พยายามศึกษาหาหนทางการแก้ไขต่อไป เช่น การนำ plugin เข้ามาช่วย หรือการลดขนาดไฟล์ ซึ่งอาจทำให้แก้ไขปัญหาไฟล์ภาพสามมิติที่มีขนาดใหญ่เกินไปได้

2) ใช้โปรแกรมพรีเฟบเข้ามาช่วยในการทำงาน โดยช่วยในเรื่องของการนำพื้นผิวเข้าไปแปะบนวัตถุ

6.4 แนวทางการพัฒนาต่อ

1) เพิ่มเฟอร์นิเจอร์ชนิดต่างๆ ให้มีรูปแบบที่หลากหลาย และมีความสวยงามมากขึ้น
2) จะทำให้แอพพลิเคชั่นสามารถแสดงอาคารในมุมมองสามมิติ และบริเวณโดยรอบสถานที่ได้
3) ให้ผู้ใช้สามารถอัพโหลดรูปเฟอร์นิเจอร์ของตัวเอง เพื่อใช้ออกแบบได้
4) แอพพลิเคชั่นสามารถประมวลผลห้องที่ผู้ใช้ออกแบบมาเป็นภาพในมุมมองต่างๆ ให้ผู้ใช้สามารถบันทึกได้

บรรณานุกรม

Rob Bateman and Richard Olsson. 2010. **The Essential Guide to 3D in Flash**. New York : friendsof

Michael Lively. 2010. **Professional Papervision3D**. Great Britain : Wrox.

Keith Peters. 2007. **Foundation ActionScript 3.0 Animation : Making Things Move!**. USA : friendsof

Jeff Winder, Paul Tondeur. 2009. **Papervision3D Essentials**. Birmingham : PACKT Publishing

Dariush Derakhahani, Randi Lorene Munn. 2008. **Introducing 3ds Max 2008**. Canada : Autodesk

บรรณาธิการ. 2551. **สร้างงาน 3d Animation กับ 3dMax 2009**. นนทบุรี : Digiart

วิกิพิเดีย. 2552.**COLLAD**.[Online].Available : http://en.wikipedia.org/wiki/COLLADA