

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

เกมต่อสู้ของหุ่นยนต์โดยการโปรแกรมในระบบออนไลน์

J-ROBOT BATTLE PROGRAMMING ONLINE



นายพนพร ศิริชนะวัฒนา
นายอภิชาติ สุขศรี

เลขหมู่.....
เลขทะเบียน 61329
วัน,เดือน,ปี 17 ก.ค. 2549

b.....
i.....

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2547

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เกมต่อสู้ของหุ่นยนต์โดยการโปรแกรมในระบบออนไลน์

J-ROBOT BATTLE PROGRAMMING ONLINE



ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2547

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปริญญานิพนธ์ ปีการศึกษา 2547

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง เกมต่อสู้ของหุ่นยนต์โดยการโปรแกรมในระบบออนไลน์

J-ROBOT BATTLE PROGRAMMING ONLINE

คณะผู้จัดทำ นายณพพร ศิริชนะวัฒนา รหัสประจำตัว 45015370

นายอภิชาติ สุขศรี รหัสประจำตัว 45015396



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เกมต่อสู้ของหุ่นยนต์โดยการโปรแกรมในระบบออนไลน์

นายณพพร ศิริชนะวัฒนา 45015370

นายอภิชาติ สุขศรี 45015396

ดร.อรัญญา วลัยรัชต์ อาจารย์ที่ปรึกษา

อาจารย์ธนา หงษ์สุวรรณ อาจารย์ที่ปรึกษา

ปีการศึกษา 2547

บทคัดย่อ

โครงการนี้นำเสนอโปรแกรมคอมพิวเตอร์ประเภทเกมในลักษณะของ Online Game Programming 2D ผ่านทางเว็บเบราว์เซอร์ โดยผู้เล่นเป็นผู้กำหนดท่าทาง และขั้นตอนการต่อสู้ของตัวละครในเกม ท่าทางและขั้นตอนการต่อสู้จะแสดงในลักษณะของรูปภาพเพื่อช่วยให้ผู้เล่นเข้าใจได้ง่าย นอกจากนั้นยังสามารถเลือกปรับท่าทางของแต่ละการต่อสู้ได้อีกด้วย ค่าพารามิเตอร์ต่างๆจะถูกส่งไปยังเซิร์ฟเวอร์เพื่อทำการประมวลผลการเล่นกับคู่ต่อสู้อีกฝ่ายหนึ่ง

ในส่วนของการแสดงผลการเล่นจะถูกส่งมาแสดงผ่านเว็บเบราว์เซอร์ของผู้เล่น เกมยังเพิ่มความสนุกสนานและน่าสนใจ โดยให้มีผู้เล่นหลายคนเข้ามาร่วมแข่งขันกัน และมีการจัดอันดับของคะแนนเปรียบเทียบระดับผู้เล่นแต่ละคน

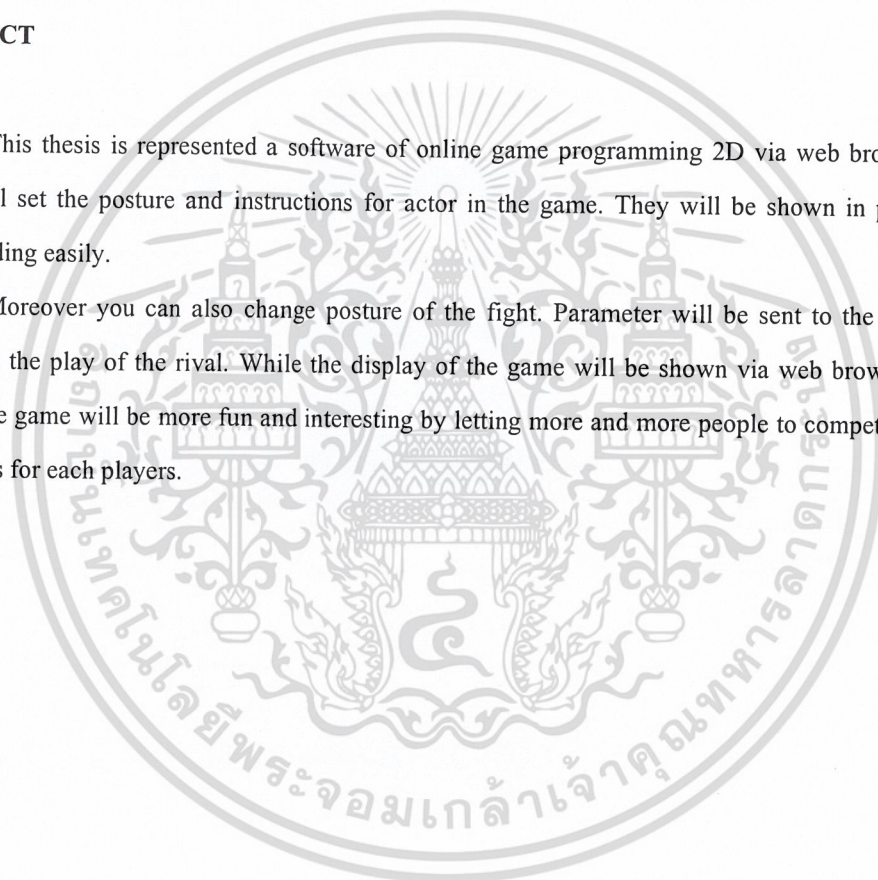
J-ROBOT BATTLE PROGRAMMING ONLINE

Mr.Nopporn	Siritanawatana	
Mr.Apichat	Suksri	
Dr.Aranya	Walairacht	Advisor
Mr.Thana	Hongsuwan	Advisor
Academic Year 2004		

ABSTRACT

This thesis is represented a software of online game programming 2D via web browser. The player will set the posture and instructions for actor in the game. They will be shown in picture for understanding easily.

Moreover you can also change posture of the fight. Parameter will be sent to the server for processing the play of the rival. While the display of the game will be shown via web browser of the player. The game will be more fun and interesting by letting more and more people to compete and will rank points for each players.



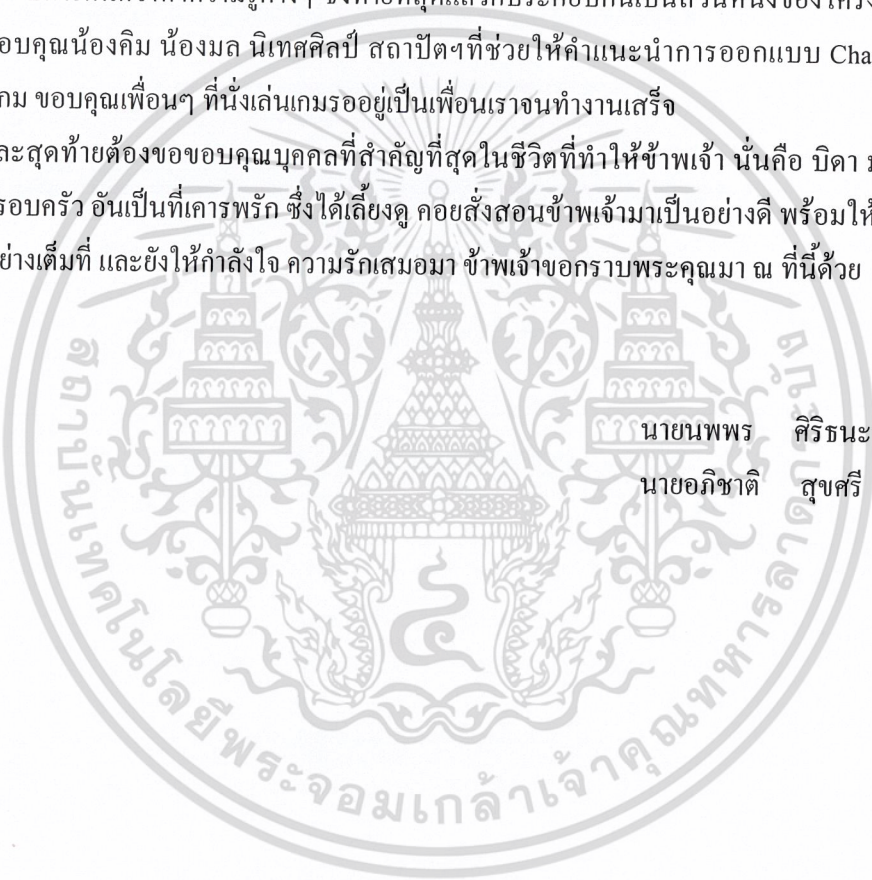
กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงได้ดีด้วยคำแนะนำ คำปรึกษาและคอยดูแลจากหลายๆฝ่ายด้วยกัน โดยเฉพาะอาจารย์ที่ปรึกษาทั้งสองท่านที่ให้โอกาสให้ข้าพเจ้าได้ทำปริญญานิพนธ์ฉบับนี้ คอยให้ความเอาใจใส่ แนะนำและความช่วยเหลือเสมอมา คือ ดร.อรัญญา วลัยรัชต์ และอาจารย์ธนา หงษ์สุวรรณ ซึ่งต้องขอขอบพระคุณเป็นอย่างสูง

ขอขอบคุณภาควิชาวิศวกรรมคอมพิวเตอร์ที่ได้จัดเตรียมสิ่งอำนวยความสะดวก เพื่อให้การวิจัยและพัฒนาโปรแกรมเป็นไปได้ด้วยความสะดวกและรวดเร็ว รวมทั้งยังมีอินเทอร์เน็ตความเร็วสูงให้บริการ สำหรับการค้นคว้าหาความรู้ต่างๆ ซึ่งท้ายที่สุดแล้วก็ประกอบกันเป็นส่วนหนึ่งของโครงการนี้

ขอบคุณน้องกิม น้องมล นิเทศศิลป์ สถาปัตย์ที่ช่วยให้คำแนะนำการออกแบบ Character ตัวการ์ตูนในเกม ขอขอบคุณเพื่อนๆ ที่นั่งเล่นเกมรออยู่เป็นเพื่อนเราจนทำงานเสร็จ

และสุดท้ายต้องขอขอบคุณบุคคลที่สำคัญที่สุดในชีวิตที่ทำให้ข้าพเจ้า นั่นคือ บิดา มารดาและบุคคลในครอบครัว อันเป็นที่เคารพรัก ซึ่งได้เลี้ยงดู คอยส่งสอนข้าพเจ้ามาเป็นอย่างดี พร้อมให้โอกาสในการศึกษาอย่างเต็มที่ และยังให้กำลังใจ ความรักเสมอมา ข้าพเจ้าขอกราบพระคุณมา ณ ที่นี้ด้วย



นายณพพร ศิริชนะวัฒนา
นายอภิชาติ สุขศรี

สารบัญ

	หน้าที่
บทคัดย่อภาษาไทย	I
บทคัดย่อภาษาอังกฤษ	II
กิตติกรรมประกาศ	III
สารบัญ	IV
สารบัญภาพ	VI
บทที่ 1 บทนำ	
1.1 ความสำคัญและความเป็นมาของปัญหา	1
1.2 ความมุ่งหมายและวัตถุประสงค์ของปัญหา	1
1.3 ผลที่คาดว่าจะได้รับ	1
1.4 ขอบเขตการวิจัย	2
1.5 ขั้นตอนของการศึกษา	2
บทที่ 2 ทฤษฎี	
2.1 ฐานข้อมูลแบบ Relational Database	3
2.2 แอปเพลท (บนเว็บเบราว์เซอร์) สำหรับแอปพลิเคชัน	3
2.3 แอปเพลท (บนเว็บเบราว์เซอร์) สำหรับ Servlet	4
2.4 แอปพลิเคชันเซิร์ฟเวอร์	5
2.5 Applets	6
2.6 Canvas	10
2.7 การจัดการกับ event (Event Handling)	11
2.8 Swing set	17
2.9 การจัดการเกี่ยวกับภาพและกราฟิกต่างๆ	30
2.10 ระบบจัดการฐานข้อมูล JDBC (Java Data base Connection)	34
2.11 Servlets	38
2.12 Threads	48
บทที่ 3 การออกแบบและการสร้าง	
3.1 ภาพรวมของระบบ	49
3.1.1 การสร้าง Code script	50
3.1.2 ประมวลผล Code script	51

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ (ต่อ)

	หน้าที่
3.1.3 การแสดงผล (Result script)	52
3.2 Function ที่สามารถกำหนดลงไป AI ได้	53
3.3 การออกแบบฐานข้อมูล	55
3.4 การประมวลผลทางฝั่ง server	55
3.5 แผนภาพ Use case diagram	56
3.6 แผนภาพ Activities Diagram	58
3.7 แผนภาพ Collaboration diagram	59
3.8 แผนภาพ Sequence diagram	63
3.10 กฎในการเล่นเกม (Rules of game)	70
บทที่ 4 การทดลอง	
4.1 เครื่องมือที่ใช้ในการพัฒนา	71
4.2 ทำการทดลอง	72
บทที่ 5 บทสรุปและวิจารณ์	
5.1 บทสรุปและวิจารณ์	76
5.2 แนวทางในการพัฒนาต่อ	76
ภาคผนวก ก. คู่มือการติดตั้งโปรแกรม	77
ภาคผนวก ข. คู่มือการใช้งานโปรแกรม	81
ภาคผนวก ค. Character ของตัวละคร	90
บรรณานุกรม	96

สารบัญภาพ

รูปที่	หน้าที่
2.1	Servlets(ใน Servlet Engine) ทำการต่ออยู่กับ Relational Database ผ่านทาง JDBC 3
2.2	java.net.URLConnection class เป็นตัว Applet ที่เป็นเส้นทางการสื่อสารของ servlet 4
2.3	Application Server 5
2.4	แสดงการที่ Interpreter จะถูกเปลี่ยนไปเป็น api ของ platform 17
2.5	แสดงการติดต่อระหว่างโปรแกรมกับระบบฐานข้อมูล 34
2.6	แสดงสถานะแวดล้อมของ servlet 38
3.1	ภาพรวมของการสร้าง code script ที่เก็บไปยังเซิร์ฟเวอร์ 49
3.2	ภาพรวมการนำ code script มาประมวลผลแล้วส่งผลลัพธ์กลับไปยัง client 49
3.3	ภาพแสดงการ modify AI โดยกำหนดลำดับการต่อสู้ของหุ่นยนต์ 50
3.4	แสดงตัวอย่าง code script ที่ได้จากการ modify AI 50
3.5	แสดงการนำเอา code script ไปเก็บไว้ที่ server 50
3.6	แสดงการนำเอา code script แต่ละฝ่ายมาทำการประมวลผล ซึ่งได้ผลลัพธ์เป็น result script 51
3.7	แสดงการส่ง result script กลับไปยัง client เพื่อนำไปแสดงผล 51
3.8	แสดงการนำเอา result script มาแสดงผล 52
3.9	แสดงฟังก์ชัน punch และ special 54
3.10	แสดงฟังก์ชัน kick 54
3.11	แสดง ฟังก์ชัน guard 54
3.12	แสดง Schema ที่ได้ออกแบบไว้ 55
3.13	แสดงการนำเอา code แต่ละฝั่งมาประมวลผล 55
3.14	แผนภาพ Use case diagram แสดงภาพรวมของเกม 56
3.15	Activity Diagram ของเกม 58
3.16	Collaboration Diagram ของการ Register 59
3.17	Collaboration Diagram ของการ Login 59
3.18	Collaboration Diagram ของการ Create Game 60
3.19	Collaboration Diagram ของการ Join Game 60
3.20	Collaboration Diagram ของการ Modify code 61
3.21	Collaboration Diagram ของการ Simulation 62
3.22	Sequence Diagram ของการ Create Game 63
3.23	Sequence Diagram ของการ Join Game 64
3.24	Sequence Diagram ของการ Login 65

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญภาพ (ต่อ)

รูปที่	หน้าที่
3.25	Sequence Diagram ของการ Modify code 66
3.26	Sequence Diagram ของการ Register 67
3.27	Sequence Diagramของการ Simulation 68
3.27	Class Diagram 69
4.1	แสดงการภาพรวมการทดลอง 71
4.2	กำหนดค่าโดยผู้เล่นชื่อ asd 72
4.3	กำหนดค่าโดยผู้เล่นชื่อ zxc 72
4.4	แสดงผลการแข่งขันกรณีเสมอ (Draw) 73
4.5	กำหนดค่าโดยผู้เล่นชื่อ asd 74
4.6	กำหนดค่าโดยผู้เล่นชื่อ zxc 74
4.7	แสดงผล asd เป็นฝ่ายชนะ 75
4.8	แสดงผล zxc เป็นฝ่ายแพ้ 75
ก.1	แสดงโปรแกรมที่ใช้ในการสร้างฐานข้อมูล 78
ก.2	แสดงโปรแกรม Tomcat ที่ใช้เป็น web server 79
ก.3	แสดงฝั่งผู้เล่นที่เข้าไปเล่นเกม 80
ข.1	หน้าจอแสดงการกรอกชื่อและรหัสผ่าน 81
ข.2	ส่วนแสดงหน้าหลักของเกม 81
ข.3	ภาพหน้าจอแสดงการสร้างห้อง 82
ข.4	หน้าจอที่จะเห็นเฉพาะผู้สร้างห้องเท่านั้น 82
ข.5	หน้าจอที่จะเห็นโดยผู้เล่นทั่วไป 83
ข.6	หน้าจอแสดงการพร้อมการเล่นเกม 83
ข.7	หน้าจอแสดงข้อความแจ้งเพื่อการโหลดเข้าสู่หน้าเกม 84
ข.8	หน้าจอแสดงผลการ Simulate เกม 84
ข.9	แสดงหน้าจอเพื่อเข้าไปทำการ modify code 85
ข.10	หน้าจอแสดงข้อความแจ้งเพื่อการโหลดเข้าสู่หน้าเกม 86
ข.11	หน้าจอแสดงผลการรันโปรแกรม 86
ข.12	POP UP ยืนยันการ Sign out 87
ข. 13	แสดงหน้าการสมัครสมาชิก 88
ข.14	แสดงการทดสอบการเลือกชื่อ 88
ค.1	แสดงภาพโปสเตอร์ของเกม 89

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญภาพ (ต่อ)

รูปที่		หน้าที่
ก.2	แสดงภาพตัวละครทั้ง 5 ตัว	90
ก.3	แสดงท่าทางการเคลื่อนไหวและต่อสู้ของตัวละครทางด้านขวา	92
ก.4	แสดงท่าทางการเคลื่อนไหวและต่อสู้ของตัวละครทางด้านซ้าย	95



บทที่ 1

บทนำ

1.1 ความสำคัญและความเป็นมาของปัญหา

เกมส่วนใหญ่ในท้องตลาดที่ผลิตออกมาขาย ไม่ว่าจะเป็นเกมแบบออฟไลน์ หรือจะเป็นเกมออนไลน์ที่นิยมเล่นปัจจุบัน ก็ได้รับความนิยมจากเยาวชนไทยเป็นจำนวนมาก ซึ่งมักจะพบอยู่เสมอว่าเยาวชนไทยในส่วนใหญ่ มักคิดเกมจากค่ายผู้ผลิตจากเมืองนอกอย่างจอมแถม หลายคนมักเสียเวลากับการเล่นเกมเป็นเวลานานๆ โดยบางครั้งกลับไม่ได้รับประโยชน์แต่อย่างใด

นอกจากความสนุกสนาน ทั้งยังเป็นการเสียเวลาและสิ้นเปลืองเงินโดยใช่เหตุ น้อยคนนักที่จะได้รับความรู้จากตัวเกมอย่างแท้จริง ดังนั้นจึงทำให้เกิดความคิดที่ว่าอยากจะพัฒนาเกม ที่ช่วยให้ผู้เล่นได้รับมากกว่าแค่ความสนุกสนานเท่านั้น เป็นสาเหตุที่ทำให้เกิด Game Programming ขึ้นมา เพื่อช่วยสอดแทรกในเรื่องของการเขียนโปรแกรมให้กับเยาวชนไทย ให้หันมาสนใจการเขียนโปรแกรมมากขึ้น มีความสนุกกับการเขียนโปรแกรม และเป็นการแสดงให้เห็นว่าเรื่องของการเขียนโปรแกรมนั้นมันไม่ยากและน่าเบื่ออย่างที่คิด

1.2 ความมุ่งหมายและวัตถุประสงค์ของการศึกษา

1. เพื่อให้คนหันมาสนใจเขียนโปรแกรมมากขึ้น
2. ช่วยให้ผู้เริ่มหัดเขียนโปรแกรม ได้เข้าใจหลักการเขียนโปรแกรมอย่างง่ายได้ดีขึ้น รวมถึงการพัฒนาการคิดอย่างมีเหตุมีผล (Logically skill)
3. เป็นพื้นฐานเพื่อช่วยให้เกิดการพัฒนาของเกมลักษณะนี้ในประเทศไทยในระดับต่อไป ทั้งความสำคัญของการพัฒนาอุตสาหกรรมเกมและ animation กำลังเป็นยุทธศาสตร์ที่สำคัญข้อหนึ่งของแผนพัฒนาแห่งชาติฉบับปัจจุบัน

1.3 ผลที่คาดว่าจะได้รับ

1. ทำให้ผู้ที่เล่นเกมได้รับความรู้เกี่ยวกับ syntax ในการเขียนโปรแกรมอย่างง่าย
2. ทำให้ผู้ที่ไม่ชอบการเขียนโปรแกรม หันมาสนใจการเขียนโปรแกรมมากยิ่งขึ้น
3. ผู้พัฒนาโครงการได้ฝึกทักษะในการพัฒนาการเขียนโปรแกรมโดยใช้ภาษา JAVA
4. เพื่อให้กลุ่มผู้พัฒนาเกม บริษัทพัฒนาเกมต่างๆ ได้เล็งเห็นถึงความสำคัญในการพัฒนาเกมในรูปแบบ Game Programming ซึ่งจะเป็นประโยชน์ต่อผู้เล่น ให้หันมาพัฒนาแบบนี้มากขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1.4 ขอบเขตการวิจัย

ต้องการพัฒนาเกมโดยใช้ภาษา Java โดยแสดงผ่านเว็บเบราว์เซอร์ ช่วยให้สามารถเล่นเกมได้บนเครื่องพีซีที่มี virtual machine Java เวอร์ชัน 4 for higher และในการเล่นเกมนั้นผู้เล่นจะต้องควบคุมตัวละคร โดยใช้หลักการของภาษา Programming แบบง่าย ฟังก์ชันต่างๆจะถูกแสดงในลักษณะของรูปภาพร่วมกับฟังก์ชันการต่อสู้และการเคลื่อนไหว

1.5 ขั้นตอนของการศึกษา

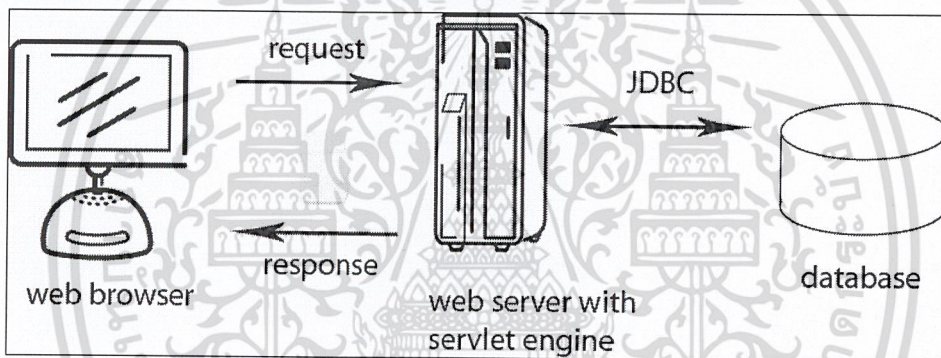
1. กำหนดขอบเขตของโครงการ
2. ศึกษาการเขียนโปรแกรมโดยใช้ภาษา JAVA ในการพัฒนาเกม 2 มิติ, เรื่องเครือข่าย และการสนับสนุนการทำงานผ่านเครือข่าย
3. วางแผนการดำเนินงาน(Project planning) และจัดเวลาในการดำเนินงาน พร้อมติดตามผล
4. ออกแบบ AI ที่มีใช้ใน Game
5. พัฒนา Module ของการสร้าง AI Script
6. พัฒนา Module ของการนำเข้า AI Script ของคู่ต่อสู้ทั้งสองฝ่ายมาประมวลผล
7. พัฒนา Module ของการนำเข้าผลลัพธ์จาก Result Script มาแสดงผล
8. ทำการออกแบบเว็บไซต์เพื่อนำ Applet (module) ที่ได้สร้างไว้ไปวางไว้บนตัวเว็บไซต์
9. สรุปผลการพัฒนาเกม และแนวทางในการวิจัยเพื่อพัฒนาต่อไป

บทที่ 2

ทฤษฎี

2.1 ฐานข้อมูลแบบ Relational Database

Database เป็นส่วนที่ถูกเพิ่มขึ้นมาใน Server Side Application โดยมักจะเป็นส่วนที่อยู่ทางท้ายสุดของระบบ ส่วนที่จะเป็นตัวเรียกใช้ Database มักจะเป็นส่วนที่ทำหน้าที่ประมวลผล request ที่มาจากผู้ใช้ซึ่งโดยทั่วไปก็คือ Servlet Engine นั่นเอง [ในกรณีของ JSP(Java Server Pages) ตัว JSP Container จะเป็นส่วนที่ทำการติดต่อกับ Database โดย JSP เองจะทำการเรียกใช้ข้อมูลต่าง ๆ โดยผ่านทาง Java Class ที่อยู่ในรูปของ JavaBean หรือ Tag Library] ตัวอย่างของ Server Side Application ที่มี Database มาเกี่ยวข้องอาจจะเป็นดังรูปข้างล่างนี้



รูปที่ 2.1 Servlets(ใน Servlet Engine) ทำการต่อกับ Relational Database ผ่านทาง JDBC

2.2 แอปเพลต (บนเว็บเบราว์เซอร์) สำหรับแอปพลิเคชัน

Applet เป็นโปรแกรมที่เขียนขึ้นโดยใช้จาวา สามารถรันที่ไหนก็ได้ที่มี JVM อยู่ แต่โดยทั่วไปตัว Applet นี้มักจะถูกรันภายใต้ JVM ที่อยู่ใน Web Browser หรือ AppletViewer ในช่วงแรก ๆ Applet ที่ใช้รันบนเว็บมักจะถูกเขียนขึ้นเพื่อใช้ตกแต่งเพจต่าง ๆ ให้มีความสวยงามแต่ด้วยความสามารถที่ว่า Applet สามารถสร้าง connection กลับไปยังเซิร์ฟเวอร์ที่เป็นตัวเก็บ Applet ดังกล่าวได้ Applet จึงถูกใส่ความสามารถอื่น ๆ เข้าไปเกินกว่าที่จะเป็นเพียง client side application แต่เพียงอย่างเดียว วิธีการแรกที่นิยมใช้กันก็คือ

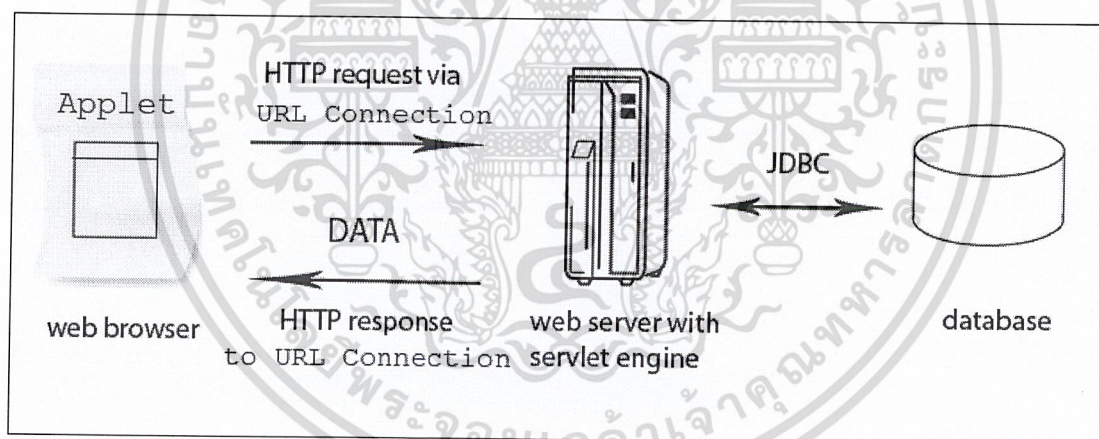
เราสามารถให้ Applet ติดต่อกับ Application ที่รันอยู่ที่ตัว Server ที่เก็บ Applet นั้นไว้โดยหลักการก็คือการใช้ web server เป็นแค่เพียงตัวเก็บ Applet เพื่อให้ web browser สามารถโหลด Applet มาจาก web server ได้โดยหลักจากที่ web browser ทำการโหลด Applet เสร็จแล้ว ตัว Applet ก็จะทำการสร้าง connection ไปยัง Application ที่รันอยู่บน Server เดียวกับที่รัน web server แทน

2.3 แอปเพลท (บนเว็บเบราว์เซอร์) สำหรับ Servlet

ในกรณีที่เว็บไซต์ต้องการความสวยงามหรือความ Interactive กับผู้ใช้อย่างสูงจนกระทั่งการใช้ HTML และ Widget ต่าง ๆ (เช่น ปุ่มหรือพวก html form) ที่ทาง web browser จัดเตรียมไว้ให้ไม่สามารถที่จะให้ความสวยงามตลอดจนความยืดหยุ่นเพียงพอต่อความต้องการของผู้ที่ออกแบบเว็บไซต์นั้นได้ Applet ก็มักเป็นอีกทางเลือกหนึ่งใช้เพื่อลดช่องว่างในส่วนนี้

ลักษณะของ Applet ที่ติดต่อกับ Servlet จะต่างจาก Applet ที่ใช้ติดต่อกับ Application ตรงที่ว่า Applet ที่ใช้ติดต่อกับ Servlet จะยังคงใช้ HTTP protocol ในการติดต่อสื่อสารกับ Server โดยถ้าเปรียบเทียบกับคือแทนที่จะให้ user ใช้ HTML Form ต่าง ๆ ของ web browser ในการติดต่อสื่อสารกับ Servlet (ที่อยู่บน web server) เราก็ใช้ Applet เป็นตัวแทนของ HTML Form แทน ซึ่งนอกจากที่ Applet จะสามารถแทนที่ HTML Form ได้แล้ว เรายังสามารถเพิ่มความสวยงามและความ interactive เข้าไปใน Applet ได้ด้วย

วิธีที่ Applet จะติดต่อกับ Servlet คือ แทนที่จะให้ web browser สร้าง HTTP Request ไปยัง Servlet เหมือนอย่างแต่ก่อนเราก็ให้ Applet ที่รันอยู่บน web browser สร้าง HTTP Request ติดต่อกับ Servlet แทนโดยวิธีการเช่นนี้เราเรียกว่า HTTP Tunnel ซึ่งสามารถทำได้ง่าย โดยให้ Applet สร้าง HTTP Connection ไปยัง Servlet ผ่านทางคลาส `java.net.URLConnection`



รูปที่ 2.2 `java.net.URLConnection` class เป็นตัว Applet ที่ป็นเส้นทางการสื่อสารของ servlet

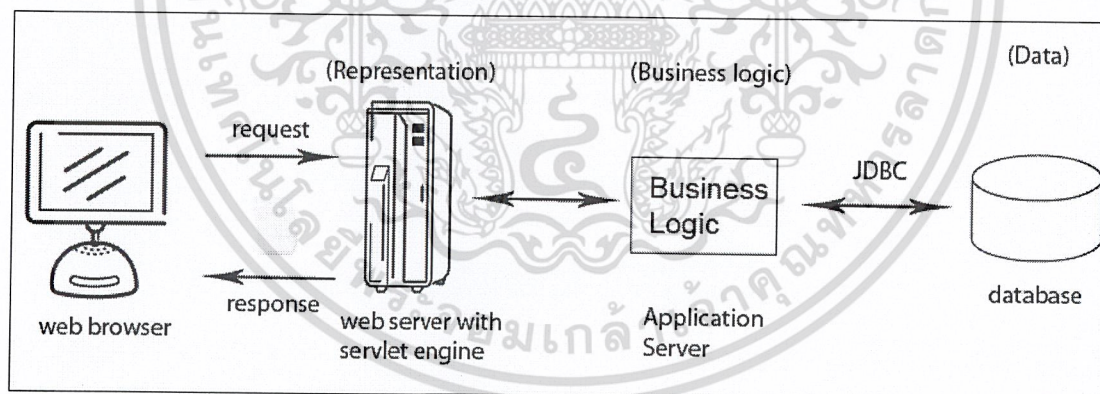
เมื่อใดที่มีการเปลี่ยนแปลงเกิดขึ้นกับ Applet ซึ่งเป็นผลมาจาก activity ต่าง ๆ ของผู้ใช้ ตัว Applet จะทำการส่งค่าที่เปลี่ยนแปลงนี้ไปยัง Servlet เพื่อทำการประมวลผล โดยหลังจากที่ Applet ได้รับผลที่ส่งไปกลับมาจาก Servlet แล้ว Applet จะทำการตีความผลที่ได้เพื่อทำการเปลี่ยนแปลงตัวมันเองโดยผลที่ได้สำหรับผู้ใช้ก็คือการเปลี่ยนแปลงของค่าตัวเลขต่าง ๆ ที่อยู่บน Applet ในกรณีที่ Applet นั้นถูกใช้สำหรับแสดงค่าจำนวนตัวเลข ตลอดจนการเปลี่ยนแปลงรูปร่างหน้าตาของ Applet ไปยัง state อื่น ๆ ในกรณีของ Applet ที่ใช้สำหรับปฏิทิน เป็นต้น

วิธีการ Applet to Servlet นิยมใช้กันมากเพราะเป็นการลดโหลดหรือการคำนวณที่หนักหน่วงจากส่วนของ client (Applet) ไปยังส่วนของ server (Servlet) แทน โดยมักจะมีศัพท์คำหนึ่งที่ใช้แทนลักษณะ architecture แบบนี้ซึ่งเราเรียกว่า thin client (นอกจาก HTTP โพรโทคอลแล้ว Applet อาจจะใช้ RMI หรือ CORBA ในการติดต่อสื่อสารกับ Servlet ที่สนับสนุน Middleware สองอันนี้ก็ได้)

2.4 แอปพลิเคชันเซิร์ฟเวอร์

Application Server เป็นส่วนประกอบที่ถูกสร้างขึ้นมาเพื่อสนองความต้องการที่ว่า ในการสร้างระบบที่ซับซ้อนหนึ่งขึ้นมานั้น เราควรจะแยกส่วนที่ทำหน้าที่เกี่ยวกับการบริการต่าง ๆ รวมไปถึงการจัดสรรข้อมูลออกมาเป็นอีกส่วนต่างหาก (Business Logic) ทั้งนี้เพื่อเพิ่มความคล่องตัวในการเปลี่ยนแปลงของระบบ รวมไปถึงการจัดระบบให้แบ่งออกเป็นส่วน ๆ อย่างชัดเจนมากยิ่งขึ้น โดยถ้าดูจากบทบาทของ Application Server]

Application Server คือส่วนที่ช่วยอำนวยความสะดวกให้กับ Server Side Application โดยมักจะเป็นตัวช่วยในการคำนวณข้อมูล(Services), ช่วยเป็นตัวกลางในการจัดส่งข้อมูล(Messaging Services), ช่วยในการควบคุมการจัดเก็บข้อมูลลงไปใน database อย่างถูกต้อง(Transaction) ตลอดจนช่วยในการเพิ่มความเร็วให้กับระบบโดยการเก็บ object ต่าง ๆ ที่ใช้หรือไม่ใช้แล้วไว้เพื่อนำมาใช้อีก(Caching) เป็นต้น (ในแง่ของการออกแบบระบบ Application Server มักจะเป็นส่วนที่เราใช้ในส่วนที่เรียกว่า Services Layer ซึ่งในจำว่าแล้วส่วนนี้ก็มักจะเกี่ยวข้องกับ Enterprise Java ต่าง ๆ เช่น EJB, JTS, JMS, JNDI)



รูปที่ 2.3 Application Server

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.5 Applets

2.5.1 Applets in Jar

ปัญหาในการส่ง applet จำนวนมากไปทำงานที่ browser ของผู้ใช้คือ การส่งไฟล์ไปใน network ที่ละไฟล์จะเสียเวลามากกว่าการส่งไฟล์ใหญ่หนึ่งไฟล์ที่เป็นผลรวมของไฟล์ย่อยทั้งหมด และให้ใช้การบีบอัดเข้าช่วยด้วย การบีบอัดไฟล์ใหญ่ๆ จะได้รับประสิทธิภาพดีกว่าการบีบอัดไฟล์เล็กๆ ดังนั้นเพื่อให้การส่ง applet ผ่าน network ไปได้เร็วขึ้น เราจึงควรรว applets เหล่านั้นบีบอัดลงเป็น jar ไฟล์นี้ไปถึง browser แล้วก็ใช้ option archives ของ APPLET tag เลือกทำงาน applets ตัวที่ต้องการเป็น jar ไฟล์นั้นได้

2.5.2 Codebase

ClassLoader คือ โปรแกรมที่ทำหน้าที่ ค้นหาไฟล์โหลดคลาส เช่น เมื่อตอนคอมพิวเตอร์คอมไพล์

คลาสๆ หนึ่ง สมมติว่าชื่อคลาส A และหากคลาส A อ่านถึงคลาส B คอมไพเลอร์ก็จะบอกให้ ClassLoader โหลดคลาส B นั้นเพื่อตรวจสอบว่ามีคลาส B นั้นถูกมองเห็นได้หรือไม่ และมีสมาชิกที่จะถูกอ้างถึงได้จริงหรือไม่ ส่วนในตอนที่เราทำงาน โดยปกติ javac.exe และ javac.exe จะใช้ ClassLoader ตัวเดียวกัน ขั้นตอนการหาคลาสของ ClassLoader ตัวนี้คือ

1. หาที่ standard classes คือ rt.jar และ i18n.jar ที่อยู่ใน jdk1.3\jre\lib ถ้าใช้ JDK หรือใน Program file\JavaSoft\JRE\1.3\lib ถ้าใช้ JRE
2. หาที่ extension classes คือใน ไดรเรทอรี jdk1.3\jre\lib\ext
3. หากไม่มี classpath กำหนดไว้จะหาใน ไดรเรทอรีที่กำลังทำงาน แต่หากมี classpath กำหนดไว้จะหาในไดเรทอรี หรือ jar ไฟล์ที่ระบุใน classpath

แต่ ClassLoader ตัวนี้จะโหลดคลาสได้เฉพาะคลาสที่อยู่ในเครื่องที่ทำงาน (Localhost) เท่านั้น นั่นคือ ตำแหน่งที่ให้หาคลาสที่ระบุใน Classpath นั้นไม่สามารถเป็น URL ไปเครื่องอื่น

Browsers ที่จะทำงาน applets ได้ต้องมี JAVA interpreter ที่ต้องมี ClassLoader ทำหน้าที่โหลดคลาสให้เช่นกัน แต่เรียกว่า Applet Classloader ซึ่งต่างจาก ClassLoader ธรรมดา ตรงที่สามารถโหลดคลาสที่อยู่ต่างเครื่องได้ ขั้นตอนการค้นหาคลาสของ Applet Classloader คือ

1. หาที่ standard classes คือ rt.jar และ i18n.jar ที่อยู่ใน JAVA interpreter ของ Browsers นั้น หรืออยู่ใน Program file\JavaSoft\JRE\1.3\lib ถ้าใช้ Pluginjre
 2. หาที่ extension classes คือใน ไดรเรทอรี Program file\JavaSoft\JRE\1.3\lib\ext
 3. หากไม่มี CodeBase กำหนดไว้ จะหาใน ไดรเรทอรี ซึ่งเป็นที่อยู่เดิม ของ applet นั้น ในเครื่องที่ให้ applet นั้นมา แต่หากมี CodeBase กำหนดไว้ จะหาใน URLs ที่ระบุใน CodeBase
- กล่าวอีกอย่างได้ว่า Applet Class loader จะใช้ CodeBase เป็นเหมือนกับ Classpath เพียงแต่จะใช้หาคลาสที่อยู่ต่างเครื่อง (เครื่องที่จะยอมให้โหลดคลาสได้ต้องมี Http Server คอยส่งคลาสที่ต้องการให้)

นอกจาก Applet จะต้องใช้ CodeBase ในการโหลดคลาแล้ว บาง Applet จะต้องโหลดไฟล์ เช่น ไฟล์เสียง หรือไฟล์ภาพ จาก website ที่ให้ HTML plete หรือ Applet นั้นมา คลาส Applet จึงมีเอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Public URL getCodeBase ();

Public URL getDocumemtBase () :

ที่ให้ URL ของโคเรกทอรีที่มี applet นั้น และ URL ของไฟล์ HTML ที่เรียก applet นั้นตามลำดับ

2.5.3 AudioClip

คลาส Applet มี methods สำหรับทำการโหลดไฟล์เสียงที่เป็นไฟล์ .au คือ

Public AudioClip getAudioClip(URL url);

Public AudioClip getAudioClip(URL url,String name);

ตัวแรกมีพารามิเตอร์ URL url ที่อ้างถึงไฟล์เสียงนั้นเลย ส่วนตัวที่สองมีพารามิเตอร์ตัวแรกคือ URL url ที่อ้างถึง directory path ส่วนต้นของไฟล์เสียง และพารามิเตอร์ตัวที่สองคือ String name เป็น directory path ที่อ้างต่อจาก url นั้นไปถึงไฟล์เสียงนั้น ผลลัพธ์ที่ได้เป็น AudioClip ของไฟล์เสียงที่ถูกโหลด

AudioClip เป็น interface อยู่ใน package java.applet ผู้ผลิต browser จะทำการ implements interface นี้ให้มี methods สำหรับเล่นไฟล์เสียงนั้นให้เกิดเป็นเสียงออกมา ได้แก่

Public abstract void play();

Public abstract void loop();

Public abstract void stop();

เราใช้ play() เล่นไฟล์เสียงนั้นเพียงรอบเดียว ส่วน loop() ใช้เล่นไฟล์เสียงนั้นแบบต่อเนื่อง คือเมื่อจบแล้วจะขึ้นรอบใหม่ไปเรื่อย ๆ จนกว่าจะถูกสั่งให้หยุด โดยเรียกว่า stop()

2.5.4 Image

ในคลาส Applet มี methods สำหรับโหลดไฟล์รูปภาพคือ

Public Image getImageClip(URL url);

Public Image getImageClip(URL url,String name);

ซึ่งจะได้ reference ของ image ของไฟล์รูปภาพนั้น นั่นคือไฟล์ภาพที่ถูกโหลดขึ้นมาแล้วจะถูกเก็บเป็น instance ของคลาส Image ดังกล่าวคลาส Image อยู่ใน package java.awt

คลาส Graphics มี drawImage() เป็น method สำหรับวาด image ลงบน graphics นั้น แต่ drawImage() ถูก overload ไว้หลายตัว ตัวที่เราจะใช้โดยทั่วไปคือ

Public abstract Boolean drawImage(Image i, int x, int y, ImageObserver obs);

ซึ่งจะวาด Image ที่จะมูขี้นบน int x และ int y ส่วน ImageObserver obs โดยทั่วไปเราสามารถให้ this ซึ่งเป็น reference ของ applet ที่จะวาดรูปนั้นเป็น argument ให้แก่ พารามิเตอร์นี้ได้

2.5.5 Loading an Image form a Jar

หากเรานำ applet และไฟล์รูปภาพ ใส่ลงในไฟล์ .jar เพื่อความสะดวกในการทำส่ง ดังในตัวอย่าง ที่การโหลดไฟล์รูปภาพจากไฟล์ .jar เข้าสู่ applet นั้นจะใช้ getImage() ส่วนในกรณีนี้เราต้องสร้าง URL ของไฟล์รูปภาพในไฟล์ .jar นั้นขึ้นมาก่อนโดยใช้

```
public Class getResource(String name);
```

ซึ่งเป็น method ของคลาส Class ที่จะให้ URL ของ String name ที่ระบุ สังเกตว่าคลาส Applet มี

```
public Class getClass();
```

สำหรับเรียกขอ reference ของคลาส Class เพื่อนำมาเรียก getResource() ได้ เมื่อได้ URL ของไฟล์รูปภาพนั้นแล้วก็เรียก

```
public Object getContent();
```

เพื่อข้อมูลของไฟล์ที่ระบุโดย URL นั้น ให้ผลลัพธ์ออกมาเป็น Object เราต้องทำ casting ให้กลายเป็น ImageProducer และส่งให้แก่

```
Public Image createImage(ImageProducer b);
```

เพื่อสร้างเป็น Image i ที่นำไปวาดได้โดยใช้ drawImage() ตามปกติ สังเกตว่า createImage() เป็น Method ของคลาส Component และถ่ายทอดมาสู่คลาส Applet

2.5.6 Applet as a Panel

คลาส Applet อยู่ใน package java.applet แต่คลาส Applet ขยายจากคลาส Panel ที่ขยายมาจากคลาส Container และคลาส Component ซึ่งอยู่ใน package java.awt แสดงว่า applet เป็นทั้ง panel container และ component นั่นคือ applet จะสามารถแสดงตัวเองได้เหมือนกับ component และเราสามารถวาง component อื่น ๆ ลงใน applet ได้เหมือนกันเป็น container อันหนึ่งและ applet เป็น non-top level เหมือนกับ panel หมายความว่า เราต้องนำ applet ขึ้นแสดงบน component ที่เป็น top level ที่มีอยู่ก่อน ไม่สามารถนำขึ้นแสดงเป็นตัวแรก

2.5.7 AppletContext

ใน Package java.applet มี AppletContext เป็น interface สำหรับให้ browser นำไป implements เพื่อให้ applets สามารถเรียกใช้ methods ของ interface นี้ในการติดต่อกับสภาพแวดล้อมที่ของ browser ที่ applet นั้นกำลังทำงานอยู่

ในการ implements AppletContext โดยปกติ browser มักจะต้องมี Hashtable (หรือ Vector) สำหรับเก็บ references ของ applets ทั้งหมดที่กำลังทำงานอยู่ภายใน browser นั้น เพื่อให้ applets สามารถเรียกขอ reference ของ applets อื่น และอ้างอิงกันโดยใช้

```
public Applet getApplet(String name); ให้ applet ที่มีชื่อ name
```

```
public Enumeration getApplets(); ให้ Enumeration ของ ที่ AppletContext นี้รู้จัก
```

โดยทั่วไป browser ควรมีกลไกในการโหลดไฟล์ภาพหรือเสียงอยู่แล้ว เพื่อให้ Applet เรียกใช้กลไกเอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ไวกเหล่านี้ได้โดยไม่ต้องสร้างโปรแกรมสำหรับหน้าทีนี้อีก จึงสร้างเป็น methods ไว้ใน AppletContext คือ

`public Image getImage(URL url);` สำหรับโหลดไฟล์ภาพจาก URL url

`public AudioClip getAudioClip(URL url);` สำหรับโหลดไฟล์เสียงจาก URL url

AppletContext มี methods สำหรับให้ applets นำ HTML page มาแสดงใน browser โดยใช้

`Public void showDocument(URL url);`

HTML page ที่ระบุโดย URL url จะถูกนำมาแสดงแทนที่ page ปัจจุบัน

และยังมี method สำหรับแสดงข้อความใน status bar ของ applet คือ

`Public void showStatus(String msg);`

Applet ที่กำลังทำงานอยู่ใน browser จะสามารถเรียกขอ reference ของ AppletContext ได้โดยใช้ `getAppletContext()` ซึ่งเป็น method หนึ่งในคลาส Applet เมื่อได้ AppletContext มาแล้ว ก็จะสามารถเรียกใช้ methods เหล่านี้ได้



2.6 Canvas

คลาส Canvas ขยายจากคลาส Component ใช้สร้าง simple component ที่ว่างเปล่า แต่เดิมเรานำไปสร้างเป็นคลาสสำหรับ component ต้องการได้ โดยวาด graphic ลงไปเอง แต่หลังจากมี lightweight Components ใน swing set แล้ว เราก็มักไม่ใช่ canvas ในการสร้าง component อีกต่อไปแล้ว ประโยชน์ของ canvas จึงเหลือเพียง ใช้เป็นพื้นที่สำหรับวาด graphics หรือ image ลงไป เพื่อถูกจัดการเหมือนกับรูปภาพนั้นเป็นหนึ่ง component

ปกติเราจะไม่ใช้งานคลาส Canvas โดยตรง แต่จะสร้างคลาสลูกที่มีการ override paint() ที่ทำการวาด graphic หรือ image ที่ต้องการ และควรมี getPreferredSize() เพื่อกำหนดขนาดที่ต้องการ



2.7 การจัดการกับ event (Event Handling)

2.7.1 KeyEvent

ปกติโปรแกรมที่มี GUI ใช้แล้วจะไม่ติดต่อกับผู้ใช้โดยการอินพุตแบบ text mode และถึงแม้ต้องมีการรับข้อความทาง keyboard ก็ใช้ textfield แต่อย่างไรก็ตามโปรแกรมยังต้องการให้ components สามารถรับ events จาก keyboard ได้เพราะ อาจต้องรับ control keys อย่างเช่น Home, PgDn, End หรือ arrow keys ไปควบคุมการแสดงผล รับ Tab ไปเพื่อเลื่อน focus ระหว่าง components และรับ short-out keys เพื่อเรียกให้โปรแกรมบางอย่างทำงาน

คลาส KeyEvent อยู่ใน package java.awt.event ขยายจากคลาส InputEvent สำหรับใช้สร้าง event ที่เกิดจาก keyboard เมื่อผู้ใช้กดหรือปล่อย key จะมี key event ถูกสร้างขึ้นโดย component ที่ได้ focus ในตอนนั้น คลาส KeyEvent โดยมี constructor ดังนี้

```
Public KeyEvent(Component source, int id, long when, int modifiers, int keyCode);
Public KeyEvent(Component source, int id, long when, int modifiers, int keyCode,
Char keyChar);
```

คลาส KeyEvent มีสมาชิกเป็นข้อมูลเกี่ยวกับว่า component ที่ได้ focus ตอนที่เกิด key event นั้นเป็น component ใด เกิดขึ้นที่เวลาใด มีการกด modifier key หรือไม่และอื่น ๆ อีก ดังนี้

- Int ระบุว่า key event นั้นเป็นประเภทใด มีอยู่ 3 แบบ กำหนดเป็นค่าคงที่ไว้ในคลาสนี้ ดังนี้

public static final int KEY_PRESSED;	เมื่อ key ถูกกด (มีค่าเป็น 401)
public static final int KEY_RELEASED;	เมื่อ key ถูกปล่อย (มีค่าเป็น 402)
public static final int KEY_TYPED;	เมื่อ key ถูกกดถูกปล่อย (มีค่าเป็น 400)

หาก keyboard นั้นมีระบบ auto-repeat การกด key ค้างไว้จะทำให้มี key event ที่เป็น KEY_PRESSED ถูกสร้างขึ้นทุกครั้งที่มีการทำ key repeated จึงเป็นไปได้ที่อาจมี KEY_PRESSED หลายครั้งก่อนมี KEY_RELEASED

เฉพาะ Caps Lock กับ Num Lock จะถูกจัดการต่างจาก key อื่น คือ key event จะถูกสร้างขึ้นเมื่อปุ่มนั้นเปลี่ยนสถานะเท่านั้น เช่น เมื่อปุ่ม Caps Lock ถูกกดและปล่อย จะมี key event ที่เป็น KEY_PRESSED ของปุ่มนี้ถูกสร้างขึ้นเพื่อบอกว่าปุ่ม Caps Lock ตอนนี้ถูก enable แล้ว หลังจากนั้นเมื่อปุ่ม Caps Lock ถูกกดและปล่อยอีกครั้ง จะมี key event ที่เป็น KEY_RELEASED ของปุ่มนี้ถูกสร้างขึ้นเพื่อบอกว่าปุ่ม Caps Lock ตอนนี้ถูก disable แล้ว

ในคลาส KeyEvent นี้ยังมีค่าคงที่

```
Public static final int KEY_FIRST; มีค่าเป็น 400 แทนค่าแรกของ key event id
Public static final int KEY_LAST; มีค่าเป็น 402 แทนค่าสุดท้ายของ key event id
```

- keyChar เป็นค่า Unicode ของ key ที่ถูกกดหรือปล่อย สังเกตว่า key หนึ่งอาจมี keyChar ที่แตกต่างกันได้ ขึ้นกับว่าในตอนนั้นมี modifier key กดอยู่หรือไม่ เช่น หากปุ่ม A ถูกกด จะได้ keyChar เป็น 97 เป็นรหัสของตัว 'a' แต่ถ้าปุ่ม A ถูกกดพร้อมกับ Shift จะได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

keyChar เป็น 65 เป็นรหัสของตัว 'A' เราสามารถเรียกดู keyChar ของ key event ได้โดยใช้

```
public char getKeyChar();
```

- keyCode เป็นค่ารหัสของ key ใน keyboard ส่วนใหญ่จะถูกระบบ AWT กำหนด รหัสที่ไม่เหมือนใครไว้แล้ว แต่ละ key จะให้ keyCode เดิมเสมอไม่ว่าในตอนนั้นมี modifier key กดอยู่หรือไม่ เราสามารถเรียกดู keyCode ของ key event ได้โดยใช้

```
public int _getKeyCode();
```

ค่า keyCode ถูกกำหนดไว้เป็นค่าคงที่ในคลาสนี้ โดยมี VK_ นำหน้าชื่อ
ค่าคงที่ Alphanumeric Keys มีดังนี้

```
Public static final int VK_A, VK_B, VK_C, VK_D, VK_E, VK_F, VK_G, VK_H, VK_I,
VK_J, VK_K, VK_L, VK_M, VK_N, VK_O, VK_P, VK_Q, VK_R, VK_S, VK_T, VK_U,
VK_V, VK_W, VK_X, VK_Y, VK_Z, VK_SPACE, VK_0, VK_1, VK_2, VK_3, VK_4, VK_5,
VK_6, VK_7, VK_8, VK_9, VK_NUMPAD0, VK_NUMPAD1, VK_NUMPAD2,
VK_NUMPAD3, VK_NUMPAD4, VK_NUMPAD5, VK_NUMPAD6, VK_NUMPAD7,
VK_NUMPAD8, VK_NUMPAD9,
```

ค่าคงที่ Control Keys มีดังนี้

```
public static final int VK_BACK_SPACE, VK_ENTER, VK_ESCAPE, VK_TAB;
```

ค่าคงที่ Modifier keys มีดังนี้

```
public static final int VK_ALT, VK_CAPS_LOCK, VK_CONTROL, VK_META,
VK_SHIFT;
```

ค่าคงที่ Function Keys มีดังนี้

```
Public static final int VK_F1, VK_F2, VK_F3, VK_F4, VK_F5, VK_F6, VK_F8, VK_F9,
VK_F10, VK_F11, VK_F12, VK_PRINTSCREEN, VK_SCROLL_LOCK, VK_PAUSE,
VK_DELETE, VK_INSERT, VK_PAGE_DOWN, VK_PAGE_UP, VK_DOWN, VK_LEFT,
VK_RIGHT, VK_UP, VK_END, VK_HOME, VK_ACCEPT, VK_NUM_LOCK,
VK_CANCEL, VK_CLEAR, VK_CONVERT, VK_FINAL, VK_HELP, VP_KANA,
VK_KANJI, VK_MODECHANGE, VK_NONCONVERT;
```

ค่าคงที่ Punctuation Keys มีดังนี้

```
Public static final int VK_ADD, VK_BACK_QUOTE, VK_BACK_SLASH,
VK_CLOSE_BRACKET, VK_COMMA, VK_DECIMAL, VK_DIVIDE, VK_EQUALS,
VK_MULTIPLY, VK_OPEN_BRACKET, VK_PERIOD, VK_QUOTE, VK_SEMICOLON,
VK_SEPARATER, VK_SLASH, VK_SUBTRACT;
```

แม้ว่ามีรหัส key กำหนดไว้เป็นจำนวนมาก แต่บาง keyboard ก็ยังไม่มีบาง key หรืออาจจะมีบาง key ที่ไม่ได้ถูกกำหนดไว้ใน AWT หากมีการกด key เช่นนี้จะได้ keyCode เป็น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```
Public static final int VK_UNDEFINED;
```

และ keyChar เป็น

```
Public static final int CHAR_UNDEFINED
```

Modifiers เป็นสถานะของ modifier key ขณะมีการกด key นั้น ซึ่งเราสามารถตรวจสอบสถานะของแต่ละ modifier key ได้โดยวิธีที่กล่าวใน InputEvent

2.7.2 KeyListener

KeyListener เป็น interface ที่ listener ต้องทำการ implements เพื่อจัดการกับ key events

KeyListener อยู่ใน package java.awt.event ถูกกำหนดไว้ ดังนี้

```
Public abstract interface KeyListener extends EvenListener{
    Public abstract void keyPressed(KeyEvent e);
    Public abstract void keyReleased(KeyEvent e);
    Public abstract void keyType(KeyEvent e);
}
```

เมื่อ listener ทำการ implements KeyListener และ addKeyListener() ให้ source รู้จักแล้วหากมี key event เกิดขึ้นถ้ามีค่า id เป็น

KEY_PRESSED	จะถูกส่งไปที่	keyPressed()
KEY_RELEASED	จะถูกส่งไปที่	keyReleased()
KEY_TYPED	จะถูกส่งไปที่	keyTyped()

2.7.3 ActionEvent and ActionListener

คลาส ActionEvent ขยายจากคลาส AWTEvent ถูกเก็บไว้ใน package java.awt.event มี constructor ดังนี้

```
Public ActionEvent(Object source, int id, String command);
```

```
Public ActionEvent(Object source, int id, String command, int modifiers);
```

โดย Object source เป็น object ที่สร้าง action event นั้นขึ้นมา เหตุที่ต้องให้ source เป็น Object เพราะ menu item เป็น MenuComponent ไม่ใช่ Component ส่วน int id เป็นประเภทของ action event นั้น มีค่าที่เป็นไปได้เพียงตัวเดียวคือ

```
Public static final int ACTION_PERFORMED;
```

ซึ่งอยู่ในคลาส ActionEvent นี้ ค่า id บอกเพียงว่ามี action event เกิดขึ้นเท่านั้น จึงไม่ถูกใช้ประโยชน์ เนื่องจากมี string เรียกว่า action command ที่บอกข้อมูลเกี่ยวกับ action event นั้น ข้อความใน Action command จะขึ้นกับ component ที่สร้าง action event นั้น โดย default จะเป็นดังนี้

Source	Command
Button	Button's label
List	List's selected item
MenuItem	Menu item's label
TextField	Contents of the text field

เราสามารถเรียกดู action command ได้โดย

```
Public String getActionCommand();
```

นอกจากนี้ยังมี int modifiers ที่ระบุไว้ในขณะที่เกิด action event นั้นมีการกด modifier key หรือไม่ เช่น ขณะที่ใช้ mouse กด button อาจกด Shift key ด้วย เราสามารถเรียกสถานะของ modifier keys จาก action event ได้โดย

```
Public int getModifiers();
```

แล้วตรวจสอบสถานะของแต่ละ key โดยการ and(&) กับ mask ของ key นั้น keys ที่เป็น modifiers มี 4 keys คือ Alt, Ctrl, Meta และ Shift ซึ่ง mask ของแต่ละ key ถูกกำหนดไว้ในคลาส ActionEvent ด้วย ดังนี้

```
Public abstract interface ActionListener extends EventListener {
```

```
    Public abstract void actionPerformed(ActionEvent e); }

```

และเมื่อทำการ addActionListener() ให้ source รู้จักแล้ว หากมี action events เกิดขึ้น actionPerformed() ก็จะถูกเรียก

2.7.4 TextField

คลาส TextField ขยายจาก TextComponent ใช้สำหรับสร้าง component ที่แสดงและรับ editable text จากผู้ใช้ได้ โดยผู้ใช้สามารถทำการ edit ได้ดังที่กล่าวใน TextComponent มีข้อจำกัดเพียงว่า text field จะมีเพียงหนึ่งบรรทัดเท่านั้น คลาส TextField มี constructors ดังนี้

```
Public TextField();
```

```
Public TextField(String text);
```

```
Public TextField(int columns);
```

```
Public TextField(String text, int columns);
```

พารามิเตอร์ String text จะเป็นข้อความที่ปรากฏใน text field นั้นเมื่อเริ่มต้น หากไม่มี String text จะได้ข้อความเป็น string ที่ว่างเปล่า ส่วนพารามิเตอร์ int columns นั้น ดังนั้นจึงเป็นขนาดโดยประมาณเท่านั้น ไม่สามารถรับประกันได้ว่าจะแสดงตัวอักษรได้ตามจำนวนที่ต้องการเสมอไป หากไม่มีพารามิเตอร์ int columns จะได้ค่า default ขึ้นกับ platform แต่โดยทั่วไปจะใหญ่พอแสดง String text นั้น หากมีทั้งพารามิเตอร์ String text และ int columns ค่า columns จะกำหนด preferred size แต่หาก text นั้นยาวกว่า preferred size ก็จะมี text แสดงใน text field นั้นเฉพาะส่วนหน้าที่ยังแสดงได้ ส่วนที่เหลืออาจถูกเลื่อนขึ้นมาดูได้โดยใช้ arrow keys

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เราสามารถกำหนดและเรียกดูค่า preferred size และ minimum size ของ text field ได้โดย

```
Public Dimension getMinimumSize(int columns);
```

```
Public Dimension getMinimumSize();
```

```
Public Dimension getMinimumSize(int columns);
```

```
Public Dimension getPreferredSize();
```

Methods เหล่านี้ให้ขนาดเป็น Dimension ซึ่งถ้าบอกในการคำนวณเป็นความกว้างก็ตัวอักษร จึงมี methods สำหรับกำหนดและเรียกดูค่า preferred size ของ text field เป็นจำนวนตัวอักษร คือ

```
Public void setColumns(int columns);
```

```
Public int getColumns();
```

บาง text field อาจรับข้อมูลที่เป็นความลับ เช่น password เราไม่ต้องการให้ข้อมูลที่ผู้พิมพ์ลงไป ปรากฏใน text field นั้น แต่เพื่อให้ผู้ใช้ทราบจำนวนตัวอักษรที่ถูกพิมพ์ จึงให้ตัวอักษรที่ถูกกำหนดเป็น echo character ปรากฏขึ้นแทน เท่ากับจำนวนตัวอักษรที่ถูกพิมพ์ใน text field นั้น text field นั้น จะมีข้อมูลตามที่ได้รับแต่จะปิดบังไว้โดยแสดงเป็น echo characters ทั้งหมด เรากำหนดให้ text field หนึ่งมี echo character เป็น char c ได้โดยเรียก

```
Public void setEchoChar(char c);
```

เราสามารถตรวจสอบว่า text field นั้นมีการกำหนด echo character ไว้หรือไม่โดย

```
Public Boolean echoCharIsSet();
```

แต่ถ้า text field ไม่มี echo character ผลลัพธ์ของ getEchoChar() จะไม่มีความหมาย

ในขณะที่ text field หนึ่งได้ focus อยู่ เมื่อผู้ใช้กด Enter key จะทำให้ action() ของ text field (ซึ่งได้รับถ่ายทอดมาจากคลาส Component) ถูกเรียกให้ทำงาน โดย target ของ event ที่เป็นพารามิเตอร์นั้นจะเป็น reerence ไปที่ text field ที่ถูกกด Enter key นั้น และพารามิเตอร์ Object จะเป็น string ซึ่งเป็นข้อความทั้งหมดใน text field นั้น หรือเราอาจนำ string นี้ออกมาจาก ((TextField)e.target).getText() ก็ได้

2.7.5 Scrollbar

คลาส Scrollbar ขยายจาก Component และทำการ implements Adjustable interface ใช้สำหรับสร้าง GUI ที่ให้ผู้ใช้เลือกค่าใดค่าหนึ่ง ซึ่งเป็นเลขจำนวนเต็มในช่วง (range) ที่ต่อเนื่องกำหนดโดย minimum และ maximum เราสามารถเลือกกำหนดให้ scrollbar ปรากฏในแนวตั้งหรือแนวนอนก็ได้ รูปร่างของ scrollbar จะขึ้นกับ platform แต่โดยทั่วไปมีส่วนประกอบดังนี้ ที่ปลายทั้งสองของ scrollbar มีปุ่ม arrow ให้ผู้ใช้ click เพื่อเลื่อนไปในทิศของ arrow นั้นหนึ่ง unit หรือ line ต่อการ click หนึ่งครั้ง ค่าของ scrollbar จะขึ้นอยู่กับตำแหน่งของ scrollbar box หรือเรียกว่า thumb ซึ่งเป็นกล่องสี่เหลี่ยมที่ผู้ใช้สามารถ drag เพื่อเลื่อนไปมา ระหว่างค่า minimum กับ maximum ขนาดของ scrollbar box ถูกควบคุมโดยค่าที่เรียกว่า visible amount เมื่อ scrollbar box ถูกเลื่อนไปที่ตำแหน่งใหม่จะเกิด event ของ scrollbar และตำแหน่งนั้นจะเป็นค่าของ scrollbar หากผู้ใช้ click ในที่ว่างระหว่าง scrollbar box กับปุ่ม arrows จะถูกเปลี่ยนเป็นการเลื่อนในทิศของ arrow นั้นหนึ่ง block หรือ page ต่อหนึ่ง click

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คลาส Scrollbar มี constructors ดังนี้

```
Public Scrollbar();
```

```
Public Scrollbar(int orientation);
```

```
Public scrollbar(int orientation, int value, int visible, int minimum, int maximum);
```

พารามิเตอร์ int orientation กำหนดว่า scrollbar จะปรากฏในแนวใด หากไม่มีจะได้ค่า default เป็น VERTICAL ค่า int value กำหนดค่าเริ่มต้นของ scrollbar หากไม่มีจะได้ค่า default เป็น 0 ค่า int visible กำหนดขนาดของ scrollbar box หากไม่มีจะได้ค่า default เป็น 10 ค่า int minimum และ maximum กำหนดค่าต่ำสุดและสูงสุดของ scrollbar range หากไม่มีจะได้ค่า default เป็น 0 และ 100 ตามลำดับ นอกจากนี้ยังมีอีกสองค่าที่ไม่สามารถกำหนดได้ใน constructors คือ unit increment และ block increment หมายถึงค่าที่จะเปลี่ยนไปเมื่อมีการเลื่อนไปหนึ่ง unit หรือหนึ่ง block ค่า default เป็น 1 และ 10 ตามลำดับ ค่า minimum, value และ maximum จะต้องเป็นไปตามเงื่อนไขดังนี้

```
Minimum <= value <= maximum
```

- หาก maximum ถูกกำหนดไว้ต่ำกว่า minimum จะเปลี่ยน maximum ให้เท่ากับ minimum
- หาก value มีค่ามากกว่า maximum หรือน้อยกว่า minimum จะเปลี่ยน value ให้เท่ากับ maximum หรือ minimum ตามลำดับ
- หาก visible มีค่ามากกว่า range ระหว่าง minimum กับ maximum จะให้ scrollbar box มีขนาดเท่ากับ range นั้นแหละจะไม่สามารถเลื่อนไปมาได้

ถ้าเราสร้างเป็น scrollbar โดยใช้ constructors สองตัวแรก จะได้ค่า value, visible, minimum, maximum เป็น default แต่อาจเปลี่ยนค่าเหล่านี้ได้โดยใช้

```
Public synchronized void setValues(int value, int visible, int minimum, int maximum);
```

ค่าคงที่สำหรับ orientation ถูกกำหนดไว้ในคลาสนี้ดังนี้

```
Public static final int HORIZONTAL;
```

```
Public static final int VERTICAL
```

ถ้าจะ Set หรือ get ค่าของ orientation, scrollbar, visible amount minimum และ maximum ก็ใช้

```
Public synchronized void setOrientation(int orientation);
```

```
Public int getOrientatior();
```

```
Public synchronized void setValue(int newValue);
```

```
Public int getValue();
```

```
Public synchronized void setVisibleAmount(int newAmount);
```

```
Public int getVisibleAmount();
```

```
Public Synchronized void setMinimum(int newMinimum);
```

```
Public synchronized void setMaximum(int_newMaximum);
```

```
Public int getMinimum();
```

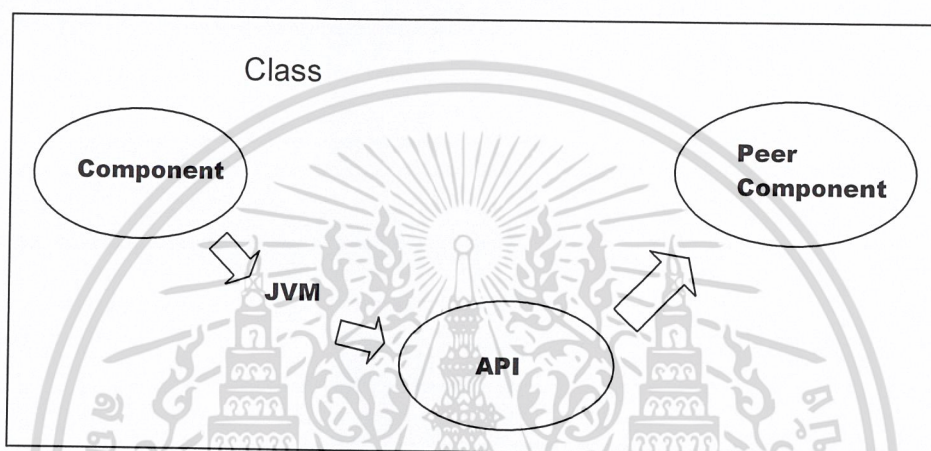
```
Public int get Maximum();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.8 Swing set

2.8.1 Heavyweight and Lightweight Components

บางคนอาจสงสัยว่า โปรแกรมของภาษา Java ไม่ขึ้นกับ platform แต่ทำไม awt set จึงขึ้นกับ platform ที่เป็นเช่นนี้เนื่องจาก interpreter ของภาษา Java ต้องเป็นของ platform ใด platform หนึ่งเท่านั้น เพราะต้องเปลี่ยน byte codes เป็นคำสั่ง api ของ platform นั้น เมื่อตอนที่โปรแกรมของคลาส Awt set ถูกคอมไพล์เป็น byte codes โปรแกรม byte codes ที่ได้จะไม่ขึ้นกับ platform แต่ตอนที่ถูกทำงานโดย Interpreter จะถูกเปลี่ยนไปเป็น api ของ platform นั้นที่ทำการวาดและควบคุมการทำงานตัว component นั้น ๆ ดังในรูปนี้



รูปที่ 2.4 แสดงการที่ Interpreter จะถูกเปลี่ยนไปเป็น api ของ platform

วิธีนี้มีข้อเสียตรงที่โปรแกรมที่ใช้คลาสของ Awt set อย่างเช่น button หากนำไปทำงานบน win32 จะ ได้ button แบบของ Windows แต่โปรแกรมเดียวกันหากนำไปทำงานบน Solaris จะได้ button แบบ Motif ทำให้ component หนึ่งเมื่อนำไปใช้งานบน platform ที่ต่างกัน จะได้รูปร่างที่ต่างกัน และที่แย่กว่านั้นคือบาง components อาจมีพฤติกรรมไม่เหมือนกัน เมื่ออยู่ต่าง platform ทำให้โปรแกรมที่ใช้งาน Awt set เหล่านั้นต้องขึ้นกับ platform

ข้อเสียอีกประการหนึ่งคือ byte codes ของ component หนึ่งเมื่อถูกทำงานจะต้องเปลี่ยนไปเป็น api จำนวนมาก เพื่อวาดและควบคุมตัว component นั้น เพราะ api หนึ่ง ๆ มักจะทำงานที่ค่อนข้างพื้นฐานมาก การติดต่อเรียกทำงานระหว่าง Java virtual machine กับ api รวมทั้งการส่ง events ให้แก่กันทำได้ค่อนข้างช้า มี overhead สูง เราจึงเรียก components ประเภทนี้ว่า heavyweight components ซึ่งหมายถึง Components ที่ต้องอาศัย api ของ platform นั้น สังเกตว่า GUIs ใน Awt set ทุกตัวเป็น heavyweight components

ทางแก้หนึ่งของปัญหาดังกล่าวคือ การใช้ lightweight components ซึ่งหมายถึง components ที่วาดตัวเองและจัดการ event ของมันด้วย Java โดยไม่ต้องใช้ api ของ platform นั้นเลย จึงทำให้รูปร่างและพฤติกรรมของมันไม่ขึ้นกับ Platform รวมทั้งปัญหาเกี่ยวกับ overhead ของ api ก็หมดไป

ตัวอย่างสุดท้ายในบทที่แล้ว แสดงการสร้าง button ที่ใช้ drawImage() วาด ของตัวเอง และมีเอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Methods สำหรับจัดการกับ event ที่เกิดขึ้นเอง แสดงว่าด้วย delegation event model ที่มีมาใน JDK1.1 นั้น เราก็สามารถสร้าง button ที่เป็น lightweight components ได้แล้ว แต่การสร้างคลาสอย่าง MyButton ในตัวอย่างนี้ค่อนข้างยุ่งยาก จึงมีคลาสสำหรับ lightweight components ให้มาใน swing set

สังเกตว่า GUIs ใน swing set ไม่ใช่ lightweight components ทั้งหมด เนื่องจาก lightweight Components ไม่สามารถใช้ Java วาดขึ้นบนจอภาพหรือ graphics devices ได้โดยตรง แต่จะสามารถวาดลงบนพื้นที่ graphics ของ component ตัวหนึ่ง ที่สร้างไว้ให้ก่อนแล้วเท่านั้น เราเรียก components ที่มีพื้นที่ Graphics สำหรับให้ components ตัวอื่นวาดว่า top-level swing components

จะเห็นว่า lightweight components ของ swing set คือพวกที่ไม่ใช่ top-level swing component ใน swing set เพียงบางตัวเป็น heavyweight components ทำหน้าที่เป็น top-level swing components ให้ lightweight components ตัวอื่น ๆ วาดใส่ตัวมัน ดังนั้นในการใช้งาน เราต้องมี top-level swing component อย่างน้อยตัวหนึ่ง ก่อนจะวาด lightweight components ลงไปได้

ขอเพียงมี top-level component วางอยู่ก่อนแล้ว ไม่ว่าจะเป็น top-level awt หรือ top-level swing ก็ตาม ก็จะวาง lightweight components ลงไปได้

แม้ว่า lightweight components ของ swing จะถูกวางไว้บน top-level awt components ได้ แต่เราไม่สนับสนุนการใช้งาน เช่น เราควรวาง heavyweight components บน top-level swing components ที่มีการสนับสนุนการทำงานของ heavyweight components ที่ดีกว่า

Jcomponent ทำหน้าที่เป็นคลาสแม่สำหรับคลาสของ lightweight components คล้ายกับคลาส Component ทำหน้าที่เป็นคลาสแม่ของ awt components

ใน swing set มีคลาสสำหรับ top-level swing components อยู่ 4 คลาสคือ JApplet, JWindow, JFrame และ JDialog แต่ละคลาสขยายมาจากคลาสแม่ของมันที่อยู่ใน awt set เพื่อใช้สร้าง applet, Window, frame และ dialog ตามลำดับ สังเกตว่า Applet awt ไม่ใช่ top-level awt component แต่ JApplet เป็น top-level swing component ความหมายของการเป็น top-level ใน awt set แตกต่างจาก swing set เล็กน้อยคือ top-level awt components คือ ตัวที่ถูกวาดขึ้นบนจอภาพหรือ graphics devices ได้เป็นครั้งแรก ซึ่งโดยนัยแล้วจะพื้นที่ graphics ให้ components ขึ้นวาดทับ ส่วน top-level swing components คือ ตัวที่พื้นที่ graphics ให้ components ขึ้นวาดทับ แต่ไม่จำเป็นต้องถูกวาดขึ้นเป็นตัวแรกบนจอภาพหรือ Graphics devices

Top-level swing components ทุกตัวมีโครงสร้างคล้ายกัน คือมี JRootPane, JGlassPane และ JLayeredPane ซึ่งกล่าวถึงในภายหลัง ในตอนนี้จะอธิบายการใช้งานเบื้องต้นของ Top-level swing Components ก่อนคือ โดยปกติเราสามารถสร้างและนำตัวเหล่านี้ขึ้นแสดงเหมือนกันตัวที่เป็นคลาสแม่ของมัน อย่างไรก็ตาม top-level swing components มีโครงสร้างที่แตกต่างไป แม้ว่าพวกมันเป็น container ด้วย แต่เราไม่สามารถ add() components ลงไปโดยตรง เราจะต้อง add() components ลงบน content pane ซึ่งเป็น Container ระดับหนึ่งใน JLayeredPane ของตัว top-level swing components นั้น ในทำนองคล้ายกัน เราไม่สามารถ setLayout() ให้แก่ตัว top-level swing components แต่ต้อง setLayout() ให้แก่ contentpane ของมัน ในคลาสของ top-level swing components เหล่านี้จะมี

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Public Container getContentPane();

ให้เรียกขอ content pane ของมัน เพื่อนำมา add() components ลงไป โดย default ทุก content pane จะมี BorderLayout ติดตั้งไว้แล้ว

2.8.2 JComponent

JComponent เป็น abstract class เป็นคลาสแม่ของคลาสสำหรับ lightweight components มีกลไกพื้นฐานของ lightweight components เช่น การทำให้ component นั้น โปร่งแสง มี Image icon, border, Tooltips, keystroke, client properties, double buffering และ debug graphics ได้ ซึ่งจะกล่าวถึงกลไกเหล่านี้ในภายหลัง จากรูปโครงสร้าง class hierarchy ที่แสดงไปก่อนหน้านี้ จะเห็นว่า JComponent ขยายออกจากคลาส Container แสดงว่า JComponent และลูกหลานของมันไม่ใช่ simple component แต่เป็น Container และเราสามารถวาง component อื่นลงในตัวมันได้ โดยใช้ add() ของคลาส Container นอกจากนั้นแต่ละตัวยังมี layout manager สำหรับจัดการขนาดและตำแหน่งของ components ที่อยู่ในตัวมันด้วย ที่จริงแล้ว jcomponent ส่วนใหญ่ไม่จำเป็นต้องทำหน้าที่เป็น container เต็มตัว เช่น label หรือ button มักไม่มี component อื่นอยู่ในตัวมัน แต่จะใช้ความสามารถในการเป็น container เก็บ icon ซึ่งจะทำให้ jlabel, JButton และ JComponent ตัวอื่น ๆ ด้วย ไม่จำเป็นต้องมีข้อความที่เป็น strings อย่าง awt set อีกต่อไป แต่สามารถมีรูปภาพที่อาจเป็น image หรือ graphic ก็ได้

คลาส MyIconComp ขยายจาก JComponent มี OvalIcon I เป็น สมาชิก มี getPreferredSize() และ getMinimumSize() สำหรับกำหนดขนาดในตอนถูกจัดการโดย layout manager และมี paint() ที่เรียก paintIcon() ของ OvalIcon I ให้วาดตัวมันลงใน Graphics g ของ MyIconComp นี้ เราทดสอบโดยใช้ JApplet ที่สร้าง MyIconComp I ขึ้นแล้ว add() นำขึ้นแสดงเหมือนกับเป็น jcomponent หนึ่ง

2.8.3 ImageIcon

Awt set มีข้อยุ่งยากเกี่ยวกับการโหลด images คือ ในคลาส Applet มี getImage() สำหรับโหลด image จากใน applet แต่ในคลาสของ Frame หรือ Dialog ไม่มี getImage() จึงต้องเรียกใช้ getImage() ที่อยู่ในคลาส Toolkit ดังนั้นหากเราสร้าง panel หรือ canvas หนึ่งที่ต้องโหลด image จะต้องเลือกกำหนดว่าจะโหลดด้วยวิธีใด และจะทำให้ panel หรือ canvas นั้นไปใช้งานใน applet หรือ frame ได้เพียงอย่างเดียว หนึ่งเท่านั้น ไม่ทั้งคู่ ในการออกแบบ swing set มีการแก้ปัญหาดังกล่าวเพื่อให้ swing components สามารถโหลด images ได้โดยไม่จำกัดว่าจะถูกนำไปใช้งานในที่ใด นั่นคือจะไม่ใช้ getImage() ดังกล่าว

ที่จริงแล้วเราสามารถสร้างคลาสที่ implements icon เพื่อโหลดและวาด image ที่ต้องการ โดยให้ทำการตรวจสอบว่า swing component ของมันทำงานอยู่ใน container เช่นใด แล้วเลือกใช้วิธีโหลด Image ที่เหมาะสมได้ คลาสของ icon เช่นนี้จะมีประโยชน์และถูกเรียกใช้บ่อย ๆ ดังนั้นเพื่อความสะดวกจึงมีคลาส ImageIcon ซึ่ง implements icon ไว้ให้แล้ว มี constructors ดังนี้

Public ImageIcon();

Public ImageIcon(String);

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

Public ImageIcon(java.net,URL);
Public ImageIcon(java.net,URL, String);
Public ImageIcon(Image);
Public ImageIcon(Image);
Public ImageIcon(Image, String);
Public ImageIcon(Byte[]);
Public ImageIcon(byte[], String);

```

เราอาจจะนำชื่อไฟล์ Image ที่ถูกโหลดเป็น String หรือเป็น url ก็ได้ และอาจโหลด image จาก instance ของคลาส Image หนึ่ง หรือ byte[] ที่อ่านมาจากไฟล์ .gif หรือ .jpeg ก็ได้ ส่วน string ที่เป็นพารามิเตอร์ตัวที่สองของ constructors บางตัวนั้นเป็น comment property ของ icon นั้น

ImageIcon มี image observer สำหรับติดตามการโหลด image ของมัน เมื่อถูกเรียก constructor ให้โหลด image มันจะไม่ return กลับไปสู่ผู้เรียกจนกว่าจะโหลด image นั้นเสร็จ ทำให้โปรแกรมที่เรียกใช้ไม่ต้องมี image observer หรือ media tracker ให้ยุ่งยาก สำหรับเพื่อรอให้การโหลด image นั้นเสร็จก่อน ถึงจะดำเนินการต่อไปได้

2.8.4 ButtonGroup

ใน package javax.swing มีคลาส ButtonGroup สำหรับกำหนดให้กลุ่มของ JRadioButton, JRadioButtonMenuItem หรือ JToggleButton มีเพียงตัวเดียวเท่านั้นที่อยู่ในสถานะ selected ที่เหลือจะต้องอยู่ในสถานะ unselected ทั้งหมด คลาส ButtonGroup สามารถใช้กับคลาสอะไรก็ได้ที่เป็นลูกหลานของคลาส AbstractButton ยกเว้นคลาส JButton และ JMenuItem เนื่องจากคลาสเหล่านี้ไม่มีสถานะ selected เมื่อตอนแรกที่ Button ใน Group ถูกนำขึ้นแสดง ทุก buttons จะอยู่ในสภาพ unselected แต่เมื่อมี button ถูก selected หลังจากนั้น button group จึงทำให้มีเพียง button หนึ่ง เท่านั้น ที่ถูก selected สังเกตว่าเราไม่มีทางใช้ program unselected Button หนึ่ง เพื่อให้ button group นั้นอยู่ในสภาพ unselected ทั้งหมด

2.8.5 List

List คือ gui ที่สามารถแสดงหลายๆ list item บนพื้นที่หนึ่ง หนึ่ง list item จะถูกแสดงใน cell หนึ่ง ซึ่งเป็นพื้นที่หนึ่ง row ใน list หนึ่ง ใน package javax.swing มีคลาส JList ขยายมาจากคลาส JComponent ใช้สำหรับสร้าง jlist เป็น light weight component แตกต่างจาก list ของ awt set ตรงที่ list items สามารถมี icon ที่เป็น image ได้ และ jlist ไม่มี scroll bar ในตัวเอง ผู้โปรแกรมจะต้องติดตั้ง jscroll bar ให้หากต้องการ คลาส JList มี constructors ดังนี้

```

Public JList();
Public JList(Object[] );
Public JList(Vector);
Public JList(ListModel);

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เราสามารถกำหนด list items ให้แก่ jlist ได้โดยพารามิเตอร์ที่เป็น Object[] โดยปกติจะให้เป็น Array of strings) หรือเป็น vector หรือ listModel ก็ได้

โดย default เมื่อ jlist ถูกสร้างขึ้นจะมีความกว้างและสูงเป็น-1 นั้นหมายถึง layout manager จะนับค่า preferred size ถ้ามี และเราสามารถกำหนดขนาดความกว้างและสูงได้โดยเรียก

```
Public void setFixedCellWidth(int);
```

```
Public void setFixedCellHeight(int);
```

ความกว้างและสูงที่กำหนดด้วยพารามิเตอร์ int นั้น เป็นจำนวน pixels เราไม่อาจกำหนดความกว้างของแต่ละ cell ได้ นั่นคือทุก cell ใน jlist หนึ่งจะต้องกว้างเท่ากัน หากเราวาง jlist ลงใน containers ที่มี Layout manager ที่ไม่ใช่ preferred size ในการคำนวณขนาด อย่างเช่น ที่ภาคเหนือ ตะวันออก ได้ และตะวันตก ของ border layout การกำหนด cell width หรือ cell height ด้วย methods ข้างบนนี้จะไม่ส่งผล

2.8.6 Borders

Awt set มี insets สำหรับสร้างขอบ (borders) ของ component นั้น ผู้ใช้จะต้องกำหนดการวาง border นั้นเอง จึงยุ่งยากและไม่สะดวกในการใช้งาน ดังนั้นใน swing set จึงมี border เป็นสมาชิกหนึ่งในคลาส JComponent สำหรับวาด borders ของ jcomponent นั้น ผู้ใช้เพียงแต่สร้าง instance ของคลาส Border ที่ต้องการแล้วนำมาติดตั้งให้แก่ Jcomponent นั้น โดยใช้

```
Public void setBorder(Border);
```

ที่จริงแล้ว border สร้างขึ้นบน insets ของ jcomponents นั้น ดังนั้นการ jcomponent ใดถูกติดตั้ง Border ก็จะทำให้ insets ของมันถูก override และถ้า border ถูกกำหนดเป็น null ก็จะทำให้ insets ของมันถูกใช้ตามปกติ

ใน package javax.swing.border มี interface ที่ชื่อว่า Border สำหรับให้คลาสของ border ต้อง Implements ซึ่งมีเพียง 3 method เท่านั้น แต่การ implements Border interface โดยตรงยังค่อนข้างยุ่งยาก จึงมีคลาส AbstractBorder เป็น abstract call ที่ทำการ implements Border ไว้ให้แล้วโดยมี methods เพิ่มขึ้นเท่าที่จำเป็นแต่เป็น border ที่ว่างเปล่า ไม่มีขนาด ใช้สำหรับเป็นคลาสเพื่อขยายไปเป็นคลาส border ประเภทต่าง ๆ ซึ่งจะสะดวกกว่าการ implements Border โดยตรง ถึงแม้ว่าเราสามารถสร้างคลาสของ border ที่ต้องการขึ้นได้ แต่เพื่อความสะดวกใน package javax.swing.border มีคลาสลูกของ AbstractBorder สร้างไว้ให้แล้วคือ EmptyBorder, BevelBorder, SoftBevelBorder, EtchedBorder, MatteBorder, TitledBorder, LineBorder และยังมีคลาส CompoundBorder สำหรับนำ border หนึ่งไปซ้อนอีก border หนึ่ง

BevelBorder	Beveled edge that makes it appear raised or lowered
CompoundBorder	Combines two borders type into a new border type
EmptyBorder	A border with no appearance
EtchedBorder	Line edge that makes it appear raised or lowered
LineBorder	Line edge with specify color and thickness
MatteBorder	Draws the borders using a solid color or a tiled images
SoftBevelBorder	BevelBorder with softer edge
TitleBorder	Combines text with other border

2.8.7 Text Field and Text Area

คลาส JTextField อยู่ใน package javax.swing ขยายมาจากคลาส JTextComponent สำหรับใช้สร้าง text field เพื่อแสดงหรือ edit ข้อความเพียงบรรทัดเดียว สามารถใช้แทนคลาส TextField ใน awt set ได้เป็นส่วนใหญ่ เพียงแต่เป็น light-weight component คลาส JTextField มี constructors ดังนี้

```

Public JTextField();
Public JTextField(String);
Public JTextField(int);
Public JTextField(String, int);
Public JTextField(Document, String, int);

```

พารามิเตอร์ที่เป็น string จะปรากฏใน text field นั้นเมื่อต้น ส่วนพารามิเตอร์ที่เป็น int ระบุขนาด Column เป็นจำนวนตัวอักษรของ text field และพารามิเตอร์ที่เป็น document คือ text's content ของ Text field ที่สร้างขึ้นนั้น ถ้าไม่มีจะใช้ default document ซึ่งได้จากการเรียก createDefaultModel() หรือใช้

```

Public Boolean isEditable ();
Public void setEditable(Boolean);

```

สำหรับ get และ set ให้ jtext component นั้นถูก edit ได้หรือไม่ ซึ่งหากถูกกำหนดให้ถูก edit ได้ก็ จะสามารถใช้ methods ต่อไปนี้

```

Public void cut();
Public void copy();
Public void paste();

```

ทำให้คลาสลูกหลานของ JTextComponent สามารถทำการ cut,copy และ paste ได้

คลาส JTextField ถูกขยายออกไปเป็นคลาส JPasswordField สำหรับสร้าง text field ที่ใช้แสดง หรือ Edit ข้อความที่เป็น passwords คือ จะไม่แสดงข้อความจริงที่ถูก edit ลงไป แต่จะปรากฏแต่อักขระที่กำหนดไว้เท่านั้น

คลาส JTextArea อยู่ใน package javax.swing สำหรับแสดงหรือ edit ข้อความหลาย ๆ บรรทัด เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คล้ายกับคลาส TextArea ของ awt set แต่ jtext area เป็น light weight component และไม่มี scroll bars โดยอัตโนมัติในตัวเอง แต่ไม่เป็นปัญหาเพราะสามารถเพิ่ม jscroll pane ให้แก่ jcomponents ได้โดยไม่ยาก คลาส JTextArea มี constructors ดังนี้

```
Public JTextArea();
Public JTextArea(String);
Public JTextArea(int, int);
Public JTextArea(String, int, int);
Public JTextArea(Document);
Public JTextArea(Document, String, int, int);
```

พารามิเตอร์ที่เป็น string จะปรากฏใน text area นั้นเมื่อต้น ส่วนพารามิเตอร์ที่เป็น int, int นั้นระบุขนาดของ text area เป็นจำนวน rows และ columns และพารามิเตอร์ที่เป็น document คือ text's content ของ text area ที่สร้างขึ้นนั้น

Option Panes

แม้ว่าใน swing set มีคลาส JDialog ที่ทำงานได้เหมือนกับคลาส Dialog ใน awt set แต่การใช้งาน Dialog เหล่านี้ เราต้องเขียนโปรแกรมเพิ่มเติม ทำให้ยุ่งยากไม่สะดวก โดยทั่วไปแล้ว dialogs มักถูกใช้ในรูปแบบที่แตกต่างกันเล็กน้อยไม่อีกแบบใน swing set จึงมีคลาส JOptionPane อยู่ใน package javax.swing เป็น utility class ซึ่งหมายถึงเราใช้คลาสนี้สร้าง jdialog หรือ jinternal frame ที่มีรูปแบบตามที่ต้องการ Jdialog หรือ jinternal frame ที่ถูกสร้างขึ้นโดยคลาส JOptionPane จะมีส่วนประกอบ 4 อย่าง

2.8.8 Table

ที่จริงแล้วเราสามารถ gui components ที่กล่าวไปแล้วสร้าง gui เพื่อแสดงข้อมูลในลักษณะสองมิติ หรือเป็นตารางได้ แต่ใน package javax.swing มีคลาส JTable สำหรับสร้าง gui เพื่อแสดงตารางซึ่งใช้งานได้ง่าย และยังให้ผู้ใช้เปลี่ยนขนาดของ row และ column ได้ในระหว่างใช้งาน

คลาส JTable ขยายจากคลาส JComponent มี constructors หลายตัว แต่จะกล่าวถึงบางตัวดังนี้

```
Public JTable();
Public JTable(int, int);
```

ตัวที่เป็น empty constructor จะสร้าง jtable ที่มี data model, column model และ selection model เป็นแบบ default ส่วนตัวที่มีพารามิเตอร์เป็น int สองตัวนั้นจะสร้าง jtable ที่มี DefaultTableModel มีจำนวน rows และ columns เป็นพารามิเตอร์ที่กำหนด โดย cells ทั้งหมดว่างเปล่า และมีชื่อของ column headers เป็น "A", "B", "C" เป็นต้น Constructors สำหรับคลาส JTable ตัวที่ใช้งานที่สุดคือ

```
public JTable(Object[] row, Object[] column);
```

พารามิเตอร์ Object[] row เป็น array สองมิติของข้อมูล (โดยปกติเป็น string) ที่จะแสดงในตาราง ส่วนพารามิเตอร์ Object[] column เป็น array หนึ่งมิติของ column headers (โดยปกติเป็น strings)

2.8.9 TableModel

หากข้อมูลที่มีไม่ได้อยู่ในรูป Array หรือ vector เราสามารถทำให้ข้อมูลนั้นถูกแสดงใน jTable ได้ โดยนำข้อมูลเหล่านั้นมาอยู่ในคลาสที่ทำการ implements TableModel ซึ่งเป็น interface ที่อยู่ใน package Javax.swing.table คลาสที่ implements TableModel จะเป็น model ให้แก่ jTable ทำหน้าที่เป็น view ด้วยวิธีนี้ เราสามารถนำข้อมูลที่ซับซ้อนมาแสดงใน jTable ได้

ใน TableModel มี methods สำหรับนำข้อมูลเหล่านั้นไปแสดงใน jTable และเนื่องจาก table model จะส่ง TableModelEvent ออกมาหากมีการเปลี่ยนแปลงเกิดขึ้นแก่ model ดังนั้นใน interface นี้จะมี Methods add() และ remove() TableModelListener ด้วย ซึ่งอาจจะทำให้การสร้างคลาสที่ implements TableModel ค่อนข้างยุ่งยาก ดังนั้นเพื่อความสะดวกใน package javax.swing.table มีการสร้าง abstract Class ที่ชื่อ abstractTableModel ไว้ให้แล้ว ซึ่งทำการ implements TableModel โดยมี methods เกี่ยวกับ TableModelEvent ไว้ให้เท่าที่จำเป็น เราสามารถสร้างคลาสลูกของ AbstractTableModel ได้โดยต้องทำการ implements เพียง 3 methods เท่านั้นคือ

```
Public int getRowCount();
Public int getColumnCount();
Public Object getValueAt(int row, int column);
```

โดย getRowCount() และ getColumnCount() จะให้จำนวน rows และ columns ของตาราง ส่วน getValueAt() จะให้ค่า object ที่แสดงใน cell ที่ระบุโดย int row และ columns นั้น

โดย default เราจะได้ column header เป็น “A”, “B” และต่อ ๆ ไป แต่หากเราต้องการกำหนด Column header เองก็ทำได้โดย override

```
Public String getColumnName(int column);
```

เมื่อสร้างคลาสของ TableModel แล้วก็นำมาสร้าง jTable ได้โดย นำ table model ส่งให้เป็นพารามิเตอร์ของ constructor ของคลาส JTable คือ

```
Public JTable(TableModel);
```

2.8.10 TableCellRenderer

เราสามารถวาง component ลงใน cell ของ jTable ได้หากคลาสของ Component นั้นทำการ Implements interface ที่ชื่อ TableCellRenderer ที่อยู่ใน package javax.swing.table ใน interface นี้มีเพียง method เดียวคือ

```
Public Component getTableCellRendererComponent(JTable table,
Object value,
Boolean isSelected,
Boolean hasFocus,
Int row, int column)
```

ซึ่งจะให้ค่าเป็น component ที่จะถูกวาง(rendered) ลงใน cell ที่ตำแหน่ง int row, int column ของเอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

JTable table ที่เป็นพารามิเตอร์ ส่วนพารามิเตอร์ Object value คือค่าของ cell ที่จะถูก rendered ซึ่งขึ้นกับการตีความของเรา เช่น ถ้าค่าเป็น string “true” เราอาจแสดงค่าเป็น string หรือจะใช้ checkbox ก็ได้ ถ้าพารามิเตอร์ Boolean isSelected เป็น true ที่ cell นั้นจะถูก rendered ด้วยสีสำหรับ selection highlighted และถ้าพารามิเตอร์ Boolean hasFocus เป็น true ที่ cell นั้นจะถูก rendered ด้วยสีสำหรับ component ที่ได้ focus

หากเราจะวาง component เช่นใดลงใน cell ของ jTable ก็สร้างคลาสลูกที่ขยายจากคลาสของ component นั้นและ implements TableCellRenderer แล้วสร้าง getTableCellRendererComponent() เพื่อกำหนดการวาง component นั้น นอกจากนี้เรายังจะต้องสร้างอีกคลาสหนึ่งเป็น model (เก็บข้อมูล) ของ component นั้น และต้องเรียก

```
Public void setDefaultRenderer(Class, TableCellRenderer);
```

ของ jTable นั้น เพื่อกำหนดให้ table cell renderer นั้นเป็น default renderer ของคลาส class ที่เป็น Model นั้น นอกจากนั้น ในคลาสที่เป็น table model เราจะต้อง override

```
Public Class getColumnClass(int column)
```

ให้ส่งค่าของ class ของคลาสที่เป็น model ของ component ที่จะถูกวางลงใน int column นั้น

2.8.11 Table Selection

นอกจากเราสามารถนำข้อมูลขึ้นแสดงใน jTable แล้ว เรายังสามารถเลือกข้อมูลที่ cells ต่าง ๆ ของ jTable ได้ด้วย คล้ายกับที่เราเลือก items ของ lists นั้นคือ table model จะส่ง ListSelectionEvent ออกไปให้ Listeners เมื่อผู้ใช้นำ model ไปเลือกที่ cells ของ jTable นั้น

โดย default แล้ว เพื่อประสิทธิภาพในการทำงาน jTable จะไม่ส่ง list selection events ออกมาเมื่อมีการเลือกที่ cells เราจะต้องเรียก

```
Public void setCellSelectionEnabled(Boolean);
```

ของ jTable เพื่อ enable ให้มีการเลือกที่ cells ของมันได้ จากนั้นต้องมีการกำหนด model ของการเลือกว่าจะเลือกแบบ cell เดียว (SINGLE_SELECTION) หรือเป็นช่วง (SINGLE_INTERVAL_SELECTION)

หรือเป็นหลาย ๆ ช่วง (MULTIPLE_INTERVAL_SELECTION) ซึ่งเป็นค่าคงที่ในคลาส ListSelectionModel

นั่นคือ ใน jTable ก็จะมี list selection model เป็น default อยู่แล้ว เราสามารถเรียกขอได้โดยใช้

```
Public ListSelectionModel getSelectionModel();
```

ของ jTable นั้น แล้วจึงใช้

```
Public abstract void setSelectionMode(int);
```

กำหนด selection mode ให้แก่ list selection model นั้น

เมื่อพร้อมแล้ว ก็ต้องทำการขึ้นทะเบียน listeners ให้แก่ list selection model โดยใช้

```
Public abstract void addListSelectionListener(ListSelectionListener);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

นั่นคือเราจะต้องมีคลาสของ listener ทำการ implements ListSelectionListener และมี valueChanged() ซึ่งมี ListSelectionEvent เป็นพารามิเตอร์ เนื่องจากการเลือกที่ cells ของ jTable จะทำให้ valueChanged() ถูกทำงาน และเราสามารถตรวจสอบได้ว่า cells ใดถูกเลือก โดยเรียก

```
Public int getSelectedRow();
```

```
Public int getSelectedColumn();
```

ของ jTable นั้น ซึ่งจะให้ค่า row และ column ของ cell ที่ถูกเลือก กรณีที่เลือกได้ cell เดียว แต่ถ้า Selection mode เป็นเลือกได้หลาย cells ก็ใช้

```
Public int getSelectedRows();
```

```
Public int getSelectedColumns();
```

ซึ่งจะให้ array ของค่า rows และ columns ของ cells ที่ถูกเลือก และถ้าต้องการทราบจำนวนของ Rows และ columns ที่ถูกเลือกก็ใช้

```
Public int getSelectedRowCount();
```

```
Public int getSelectedColumnCount();
```

2.8.12 Tress

ใน package javax.swing มีคลาส JTree ขยายจากคลาส JComponent สำหรับสร้าง gui เพื่อแสดงข้อมูลที่มีลักษณะเป็นต้นไม้ (hierarchical data) เมื่อเราสร้าง jtree แล้วก็นำขึ้นแสดงโดยการ add() ให้แก่ container ได้เหมือนกับ jcomponents ทั่วไป หรืออาจนำไปใส่ jscroll pane เพื่อให้มี scroll bar หากจำเป็น เราอาจใช้ jtree เพียงแสดงข้อมูล หรือให้ผู้ใช้เลือก nodes ซึ่งเราสามารถตรวจสอบได้ว่า node ที่ถูกเลือก หรือ path (จาก root ถึง node) ที่ถูกเลือกเป็นเช่นใด คลาส JTree มี constructors หลายตัวอย่าง เช่น

```
Public JTree();
```

```
Public JTree(Object[]);
```

```
Public JTree(Vector);
```

```
Public JTree(Hashtable);
```

ตัวที่เป็น empty constructor จะสร้าง jtree ขึ้นอย่างว่างเปล่า แล้วเราอาจนำ tree model มากำหนดให้ภายหลังได้ jtree ก็คล้ายกับ jcomponent อื่น ๆ ตรงที่มี default model ของข้อมูลที่จะถูกแสดงอยู่แล้ว โดยสร้างจากพารามิเตอร์ที่ให้แก่ constructors เช่น ตัวที่มีพารามิเตอร์เป็น Object [] ต้องเป็น Array ของ array ของ objects ซึ่ง array ย่อยจะถูกนำขึ้นแสดงเป็นหนึ่ง node และอาจมี array ย่อยเป็นส่วนประกอบได้อีก jtree ที่สร้างขึ้นแบบนี้จะไม่แสดง root node

2.8.13 T treeNode

โดยปกติ jtree จะสร้างขึ้นจาก nodes ซึ่งเป็น instances ของคลาสที่ implements TreeNode ซึ่ง เป็น interface ที่อยู่ใน package javax.swing.tree ใน interface นี้มี methods ที่ให้ข้อมูลเกี่ยวกับ node นั้น เช่น เป็น real node หรือไม่มี child nodes เท่าใด และ node ใดเป็น parent node

Nodes ที่สร้างขึ้นโดย implements TreeNode จะถูกเปลี่ยนแปลงไม่ได้ จึงไม่ค่อยใช้งานบ่อยนัก ใน Package เดียวกันนี้มี interface อีกตัวหนึ่งชื่อ Mutable TreeNode ซึ่งขยายจาก TreeNode โดยเพิ่ม Methods สำหรับให้เปลี่ยนแปลงข้อมูลของ node และสำหรับให้เพิ่ม (insert) หรือลด (delete) child nodes ให้แก่ node ได้ โดยปกติเราไม่ต้องสร้างคลาสที่ implements MutableTreeNode เอง เนื่องจากมีคลาส DefaultMutableTreeNode ที่ implements MutableTreeNode ไว้ให้แล้ว

คลาส DefaultMutableTreeNode มี constructors ดังนี้

```
Public DefaultMutableTreeNode();
Public DefaultMutableTreeNode(Object);
Public DefaultMutableTreeNode(Object, boolean);
```

Tree node ใด ๆ ยกเว้น root node ต้องมีหนึ่ง parent node และอาจมีหรือไม่มี child nodes จำนวน เท่าใดก็ได้ Empty constructor จะสร้าง tree node ที่ว่างเปล่าขึ้น โดยไม่มีทั้ง parent node และ child node สำหรับ constructor ตัวที่มี object เป็นพารามิเตอร์จะสร้าง tree node ขึ้นโดยมี object เป็นค่าที่ node นั้นแต่ ไม่มีทั้ง parent node และ child node หากใช้ constructors สองแบบที่กล่าวมา จะได้ tree node ที่ยอมให้มี parent node ได้โดย default แต่ถ้าใช้ constructor ตัวที่สาม จะมี Boolean สำหรับกำหนดว่า node นั้นจะมี child node ได้หรือไม่ คลาส DefaultMutableTreeNode มี methods สำหรับตรวจสอบ node และ สำหรับเปลี่ยนแปลง parent node หรือ child node ของ node นั้น เช่น หากต้องการนำ node หนึ่งไปเป็น on node ของอีก node หนึ่งก็ใช้ Public void add(MutableTreeNode);

2.8.14 TreeModel

คลาส JTree มี constructor อีกตัวหนึ่ง

```
Public JTree(TreeModel);
```

สำหรับสร้าง jtree ขึ้นจากคลาสที่ implements TreeModel ซึ่งเป็น interface อยู่ใน package javax.swing.tree นั่นคือ เราสามารถนำข้อมูลใด ๆ มาแสดงใน jtree(กรณีนี้ถูกใช้เป็น view) ได้โดยสร้าง เป็นคลาสของ tree model ที่มี methods สำหรับให้รายละเอียดเกี่ยวกับข้อมูลเหล่านั้นคือ

```
Public abstract Object getRoot();
Public abstract getChild(Object, int);
Public abstract getChildCount(Object, int);
Public abstract getIndexOfChildCount(Object, Object);
Public abstract Boolean isLeaf(Object);
```

หาก tree model ถูกเปลี่ยนแปลง เช่น มีการ insert หรือ remove nodes ก็จะมี instance ของคลาส เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

TreeModelEvent ถูกสร้างขึ้นและส่งไปให้ listeners ที่ขึ้นทะเบียนไว้ใน TreeModel จึงมี

```
Public abstract void addTreeModelListener(TreeModelListener);
```

```
Public abstract void removeTreeModelListener(TreeModelListener);
```

และหากมีการเปลี่ยนแปลงเกิดขึ้นกับ node ให้เส้นทางของ tree path ถึง node นั้น ก็จะทำให้

```
Public abstract void valueForPathChanged(TreePath, Object);
```

ถูกทำงาน เพื่อบอกว่ามีการเปลี่ยนแปลง tree path เกิดขึ้น

2.8.15 TreeSelectionModel

เราสามารถให้ jtree แสดงข้อมูล และให้ผู้ใช้เลือก (select) node ที่แสดงอยู่ได้ด้วยที่เป็นเช่นนี้เพราะ Jtree มีสมาชิกเป็น instance ของคลาสหนึ่งที่ implements TreeSelectionModel ซึ่งเก็บสถานะการถูกเลือกปัจจุบัน (current state of selection) ของ jtree นั้น สถานะการถูกเลือกของ jtree ถูกกำหนดโดย TreePath หรืออาจจะใช้ชุดของ Integers ก็ได้ การเปลี่ยนจาก tree path ไปเป็น integer จะถูกทำโดย instance ของ RowMapper เราสามารถทำงานได้โดยไม่ต้องมี RowMapper เพียงแต่ถ้าไม่มีจะทำให้ getSelectionRows() ให้ค่า null เรากำหนด tree selection model ให้มี mode ในการเลือกเป็นเช่นใดได้โดยใช้ค่าคงที่ที่กำหนดไว้ใน Interface TreeSelectionModel คือ

```
Public static final int SINGLE_TREE_SELECTION;
```

```
//Selection can only contain one path at a time
```

```
Public static final int CONTIGUOUS_TREE_SELECTION;
```

```
//Selection can only be contiguous
```

```
Public static final int DISCONTIGUOUS_TREE_SELECTION;
```

```
//Selection can contain any number of items that are not necessarily contiguous
```

หากไม่มี RowMapper จะทำให้ CONTIGUOUS_TREE_SELECTION เหมือนกับ DISCONTIGUOUS_TREE_SELECTION คือมี paths ที่ถูกเลือกจำนวนเท่าใดก็ได้

คลาส JTree ใช้ tree selection model ที่เป็น instance ของคลาส DefaultTreeSelectionModel ซึ่งทำการ implementation TreeSelectionModel ไว้แล้ว โดยมีกลไกที่จะบอกแก่ listeners เมื่อ path ของ การเลือกถูกเปลี่ยนแปลง เราเรียกขอ tree selection model ของ jtree ได้โดย

```
Public TreeSelectionModel getSelectionModel();
```

ซึ่งเป็น method ของคลาส JTree เมื่อได้ tree selection model ก็กำหนด mode ของการเลือกโดย

```
Public abstract void setSelectionMode(int);
```

ซึ่งเป็น method คลาส TreeSelectionModel โดยใช้ค่าคงที่ที่กล่าวมาแล้ว จากนั้นต้องทำการ addTreeSelectionListener() ให้แก่ Jtree ด้วย Listener ที่ทำการ implements TreeSelectionListener ซึ่งเป็น interface ที่อยู่ใน package javax.swing.event และมี method เดียวคือ

```
Public abstract void valueChanged(TreeSelectionEvent);
```

ซึ่งจะถูกเรียกทำงาน เมื่อมีการเลือก path ที่ jtree โดยมี tree selection event เป็นพารามิเตอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.8.16 Tree Rendering

โดย default รูป icons ของ nodes ชนิดต่าง ๆ ใน jtree จะถูกควบคุมโดย UIManager ซึ่งเป็นคลาสที่อยู่ใน package javax.swing และโดย default อีกเช่นกัน UIManager จะได้ icons มาจากระบบปฏิบัติการ แต่หากต้องการ เราสามารถเปลี่ยน icons ของ jtree ได้โดยเรียก `Public static Object put(Object, Object);`

ซึ่งเป็น static method ในคลาส UIManager เพื่อกำหนด icon สำหรับ nodes ประเภทต่าง ๆ ซึ่งแต่ละประเภทถูกกำหนดไว้เป็น string ดังนี้

"Tree.leafIcon"	icon ของ leaf nodes
"Tree.openIcon"	icon ของ nodes ที่ถูก open
"Tree.closeIcon"	icon ของ nodes ที่ถูก close
"Tree.expandedIcon"	icon ของ nodes ที่ถูก expanded
"Tree.collapsedIcon"	icon ของ nodes ที่ถูก collapsed

เราสามารถเปลี่ยน icons ของ nodes บางประเภทเท่านั้น ไม่จำเป็นต้องเปลี่ยนพร้อมกันทั้งหมด

2.8.17 TreeCellRenderer

แต่ละ node ของ jtree จะมีหนึ่ง cell เป็นพื้นที่สำหรับแสดงทั้ง icon และ text ซึ่งเราอาจกำหนด icon หรือเปลี่ยนแปลงค่าบางอย่างของ text เช่น font หรือ สีของตัวอักษรและ background ได้โดยใช้คลาสที่ Implements TreeCellRenderer ซึ่งเป็น interface ที่อยู่ใน package javax.swing.tree

โดยปกติเราไม่ต้องสร้างคลาสที่ implements TreeCellRenderer เนื่องจากใน jtree มี instance ของคลาส Default TreeCellRenderer ซึ่งเป็นคลาสที่ขยายจากคลาส JLabel ทำการ implements TreeCellRenderer อยู่ใน package javax.swing.tree มี methods สำหรับกำหนดค่าที่ใช้ในการ render Cell อย่างเช่น icon, font และสี เป็นต้น

เราสามารถเรียกขอ tree cell renderer จาก jtree ได้โดยใช้

```
Public TreeCellRenderer getCellRenderer();
```

หากเรา cast tree cell renderer ที่ได้ให้เป็น Default TreeCellRenderer ก็จะกำหนด icon, font และ color ได้โดยใช้

```
Public void setOpenIcon(Icon);
```

```
Public void setClosedIcon(Icon);
```

```
Public void setLeafIcon(Icon);
```

```
Public void setFontIcon(Font);
```

```
Public void setTextColor (Color);
```

```
Public void setTextNonSelectionColor (Color);
```

```
Public void setBackgroundSelectionColor (Color);
```

```
Public void setBackgroundNonSelectionColor (Color);
```

```
Public void setBackground (Color);
```

ซึ่งเป็น methods ของคลาส DefaultTreeCellRenderer

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.9 การจัดการเกี่ยวกับภาพและกราฟิกต่างๆ

2.9.1 Rectangles

ในวาดรูปพื้นที่ปิด (closed area) เราอาจทำได้สองแบบคือ วาดเฉพาะเส้นขอบของรูปร่าง เรียกว่า การ draw และวาดเส้นขอบพร้อมกับระบายสีภายในรูปร่าง เรียกว่าการ fill นอกจากนี้ ยังอาจวาดรูปปิดที่ถูกเสริมขอบให้มองดูเหมือนเป็นรูปสามมิติ โดยอาจกำหนดให้ดูเหมือนกับยกขึ้นหรือยุบลง เรียกว่าเป็นการ วาดแบบ 3D คลาส Graphics มี method สำหรับวาดรูปสี่เหลี่ยม ดังนี้

```
Public void drawRect(int x, int y, int width, int height);
```

```
Public abstract void fillRect(int x, int y, int width, int height);
```

```
Public void draw3DRect(int x, int y, int w, int h, Boolean raised);
```

```
Public void drawRect(int x, int y, int w, int h, Boolean reised);
```

ซึ่งจะวาดรูปสี่เหลี่ยม มีมุมซ้ายบนอยู่ที่ int x, y มีความกว้าง int width (ไปทางขวา) และมีความสูง Int height (ไปข้างล่าง) โดย drawRect() จะวาดรูปสี่เหลี่ยมที่มีแต่ขอบ ส่วน fillRect() จะวาดรูปสี่เหลี่ยมที่ถูกระบายสีภายในพื้นที่ปิดของรูป ส่วน draw3DRect() และ fill3DRect() จะวาดแบบ 3D โดยมี พารามิเตอร์ Boolean raised ถ้าเป็น true จะได้รูปสี่เหลี่ยมที่ดูเหมือนกับนูนขึ้นมา และถ้าเป็น false จะได้รูปสี่เหลี่ยมที่ดูเหมือนกับยุบลง ยังมีอีก method หนึ่งคือ Public abstract void clearRect(int x, int y, int width, int height);

ซึ่งจะลบพื้นที่สี่เหลี่ยมออกจาก int x, y และขนาด int width, height คล้ายกับการวาดรูปสี่เหลี่ยม แต่ เป็นการวาดด้วยสีที่เป็น background

2.9.2 RectangularShape and Rectangle2D

ใน package java.awt.geom มีคลาส RectangularSnape เป็น abstract class ที่ implements Shape สำหรับเป็นคลาสแม่ของคลาสที่กำหนดรูปสี่เหลี่ยมสองมิติ ในคลาสนี้ไม่ได้กำหนดข้อมูลของรูปร่างใด มีแต่เพียง methods สำหรับไว้สอบถามหรือเปลี่ยนแปลงข้อมูลของรูปสี่เหลี่ยมสองมิติที่กำหนดขึ้น

ใน package javax.awt.geom มีคลาส Rectangle2D เป็น abstract class ที่ขยายมาจากคลาส RectangularSnape สำหรับเป็นคลาสแม่ของคลาสที่กำหนดรูปสี่เหลี่ยมสองมิติ

ใน Rectangle2D มีคลาส Rectangle2D.Float และ Rectangle2D.Double เป็น static inner Classes ซึ่งทั้งสองเป็นคลาสลูกของ Rectangle2D ที่ใช้งานได้ มี constructors ดังนี้

```
Public Rectangle2D.Float();
```

```
Public Rectangle2D.Float(float x, float w, float h);
```

และ

```
Public Rectangle2D.Double();
```

```
Public Rectangle2D.Double (double x, double w, double h);
```

พารามิเตอร์สองตัวแรกกำหนดตำแหน่ง (x, y) ของมุมซ้ายบนของรูปสี่เหลี่ยมที่จะวาดเทียบกับ user

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Space และพารามิเตอร์สองตัวสุดท้ายกำหนดความกว้างและสูงของรูปสี่เหลี่ยม หากใช้ empty constructor จะได้ค่า x, y, w และ h เป็นศูนย์ ต้องกำหนดตำแหน่งและขนาดโดย Public void setRect(double, double, double, double); และ Public void setRect(Rectangle2D)

2.9.3 Transformation

Graphics2D สามารถเปลี่ยนแปลง (transform) รูปภาพที่จะวาดได้สี่แบบคือ

- translation คือการย้ายตำแหน่งของรูปภาพจากจุดหนึ่งไปอีกจุดหนึ่ง
- Rotation คือหมุนรูปภาพรอบจุดหนึ่ง
- Scaling คือการขยายหรือลดขนาดของรูปภาพ
- Shear คือการบิดรูปภาพไปในแกนใดแกนหนึ่ง

สังเกตว่าในการเปลี่ยนแปลงรูปภาพทั้งสี่แบบนี้ หากรูปเดิมที่เส้นใดที่ขนานกัน ผลที่ได้ก็จะมีเส้นเหล่านั้น ขนานกันเช่นเดิม เราเรียกการเปลี่ยนแปลงรูปภาพแบบนี้ว่า affine transformation ใน package Java.awt.geom มีคลาส AffineTransform สำหรับใช้ในการกำหนด affine transformation

คลาส Graphics2D มี instance ของ AffineTransform เป็น data member โดย default เพื่อใช้ทำ transformation ให้แก่รูปภาพที่จะถูก rendered

คลาส Graphics2D มี methods สำหรับ set และ get affine transform คือ

```
Public abstract void setTransform(AffineTransform);
```

```
Public abstract AffineTransform getTransform();
```

โดย setTransform() จะเปลี่ยน affine transform เดิมเป็นตัวใหม่ที่เป็นพารามิเตอร์ ในกรณีที่เราต้องการดัดแปลง affine transform เดิมก็ใช้

```
Public abstract void transform(AffineTransform);
```

ซึ่งจะต่อท้าย affine transform ตัวเดิมด้วยตัวที่เป็นพารามิเตอร์

คลาส AffineTransform กำหนดค่า transformation โดย 3 x 3 matrix ซึ่งจะเปลี่ยนจุด (x, y) ไปเป็น (x', y') โดยแทนจุดด้วย column vector และทำการ transform โดยคูณด้วย matrix ดังนี้

$$\begin{bmatrix} x' \\ y' \\ 1' \end{bmatrix} = \begin{bmatrix} m00 & m01 & m02 \\ m10 & m11 & m12 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} m00x + m01y + m02 \\ m10x + m11y + m12 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \\ 1' \end{bmatrix} = \begin{bmatrix} m00 & m01 & m02 \\ m10 & m11 & m12 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} m00x + m01y + m02 \\ m10x + m11y + m12 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \\ 1' \end{bmatrix} = \begin{bmatrix} m00 & m01 & m02 \\ m10 & m11 & m12 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} m00x + m01y + m02 \\ m10x + m11y + m12 \\ 1 \end{bmatrix}$$

เรากำหนดการ transform ต่าง ๆ ด้วยค่า 6 ค่าคือ m00,m01,m02,m10,m11,m12

คลาส AffineTransform มี constructors หลายตัว ให้เราสามารถกำหนดค่าในการทำ transform แบบต่าง ๆ ค่า 6 ค่า ซึ่งอาจจะเป็น float หรือ double ก็ได้ ดังนี้

```
Public AffineTransform();
```

```
Public AffineTransform(java.awt.geom, AffineTransform);
```

```
Public AffineTransform(float, float, float, float, float, float);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```
Public AffineTransform(float[]);
```

```
Public AffineTransform(double, double, double, double, double, double);
```

```
Public AffineTransform(double,[]);
```

2.9.4 Translation

คือการเลื่อนรูปภาพจากตำแหน่งหนึ่งไปอีกตำแหน่งหนึ่ง กำหนดเป็น double ค่าสองค่าคือ tx และ ty ซึ่งจะทำให้จุด x,y ใด ๆ ถูกเปลี่ยนเป็น x+ tx, y+ คลาส AffineTransform มี methods หลายตัวในการทำ translation เช่น เราสามารถ get ค่า tx และ ty ได้โดยใช้

```
Public double getTranslateX();
```

```
Public double getTranslateY();
```

ดังที่ทราบ Graphics2D มี default AffineTransform อยู่แล้ว เราสามารถเพิ่มการ translation ต่อท้ายเข้าไปใน AffineTransform ได้โดยใช้

```
Public void translate(double tx, double ty);
```

2.9.5 Fonts

ภาษา java เก็บแสดงตัวอักษรด้วยรหัส unicode ดังนั้นการ rendering text คือการวาด shape ที่กำหนดไว้คู่กับรหัส Unicode ของตัวอักษรตัวนั้น เราเรียก shape ที่กำหนดไว้สำหรับอักษรตัวหนึ่งว่า glyph แม้ว่าการ mapping ระหว่าง glyphs กับตัวอักษรส่วนใหญ่จะเป็นแบบหนึ่งต่อหนึ่ง แต่ก็มีบ้างในบางภาษาที่อักษรมากกว่าหนึ่งตัวถูก mapped ไปเป็น glyphs ตัวเดียว และอักษรบางตัวอาจ mapped ไปเป็น glyphs มากกว่าหนึ่ง ขึ้นกับตำแหน่งของตัวอักษรนั้นในคำหรือประโยค

ระบบแสดงผลส่วนใหญ่ก็มีหลาย ๆ glyphs สำหรับอักษรตัวหนึ่ง เราเรียกเซตของ glyphs ว่า font โดยปกติ font คือ glyphs ชุดหนึ่งที่กำหนดขึ้นสำหรับอักษรแต่ละตัวในภาษานั้น ซึ่งนอกจากมีข้อมูลเกี่ยวกับ glyphs แล้วยังมีข้อมูลเกี่ยวกับการ mapping ระหว่างตัวอักษรกับ glyphs ด้วย

ภาษา java มีคลาส font อยู่ใน package java.awt สำหรับใช้เก็บข้อมูลเกี่ยวกับ font ทำให้โปรแกรมสามารถเลือกใช้งาน font ที่ติดตั้งอยู่ในระบบปฏิบัติการได้ เรากำหนด font ให้แก่ component หรือ graphics เพื่อให้ใช้ font นั้นในการวาดตัวอักษร โดย

```
Public void setFont(java.awt.Font);
```

หากต้องการทราบว่า font ที่ถูกกำหนดไว้ปัจจุบันเป็น font ใด ก็เรียกดูได้โดยใช้

```
Public Font getFont()
```

Methods ทั้งสองมีอยู่ทั้งในคลาส Component และคลาส Graphics Font มีชื่อ 3 แบบ คือ

- Family Name เป็นชื่อของ fonts ตระกูลหนึ่งๆ โดยปกติ fonts ที่อยู่ในตระกูลเดียวกัน ถึงแม้จะมีขนาดหรือ style ต่างกัน ก็จะมีลักษณะ glyphs คล้ายกัน เราเรียกดู family name ของ font หนึ่งได้โดยใช้

```
Public String getFamily();
```

```
Public String getFamily(locale)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตัวที่มี locale เป็นพารามิเตอร์จะให้ family name ของ font นั้นสำหรับ locale ที่กำหนด

- Logical Name คือ ชื่อที่สมมุติขึ้นสำหรับ fonts ที่ใช้ในงานประเภทต่างๆ เช่น Dialog, Serif, SanSerif และ Monospace เป็นต้น fonts เหล่านี้ไม่มีอยู่จริงในระบบทั่วไป ในตอนทำ text rendering ชื่อ Font ที่เป็น logical name จะถูก mapped ไปเป็นชื่อ font ที่มีอยู่จริงในระบบนั้น โดยใช้ไฟล์ font.properties ที่อยู่ในไดเรกทอรี jdk\jre.lib ของ JDK1.2 ขึ้นไป(ใน jdk\lib ของ JDB1.1) หรือ \lib ของ Java Runtime Environment ไฟล์ font.properties.th ในไดเรกทอรีนี้เป็นไฟล์ที่ map ชื่อ logical name ให้เป็นชื่อ font ที่มีบางตัวเป็นภาษาไทย หากเราต้องการใช้ไฟล์นี้ ก็เปลี่ยนชื่อไฟล์จาก font.properties.th เป็น font.properties การใช้ logical font name มีมาตั้งแต่ JDK1.0 เป็นวิธีที่ทำให้โปรแกรมไม่ต้องอ้างอิงถึงชื่อ fonts ที่จะใช้งานจริง เพื่อให้โปรแกรมไม่ขึ้นกับ platform แต่มีข้อเสีย ตรงที่ logical font name ถูก mapped ไปเป็น fonts ที่ต่างกันในแต่ละ platform จึงเกิดปัญหาการ rendering text ที่ดูดีใน platform หนึ่ง แต่ไม่ดีใน platform อื่น ดังนั้นนับแต่ JDK1.2 เป็นต้นมา แม้ยังยอมให้มีการใช้ logical font name แต่ก็ไม่นับสนุนให้ใช้อีกต่อไป คลาส Font มี

```
Public String getName();
```

ซึ่งจะให้ logical font name ของ font นั้นออกมา เราสามารถตรวจสอบว่า JDK ที่ใช้นี้มี logical font name ไต่บ้าง โดยใช้

```
Public abstract String getFontList();
```

ซึ่งเป็น method ในคลาส Toolkit ซึ่งจะให้ชื่อ logical fonts ที่มีออกมาเป็น array ของ strings สังเกตว่า Sreif เป็น font คลาสสิก แบบอักษรไทยโบราณ มักจะถูก mapped เป็น Times Roman Sans Serif เป็น font ที่ดูทันสมัย และง่าย มักจะถูก mapped เป็น Arial หรือ Helvetica Monospace เป็น font ที่แต่ละตัวอักษรรูปร่างเท่ากัน เหมาะสำหรับ program codes มันจะถูก Mapped เป็น Courier

- Face Name คือชื่อ font ที่มีอยู่จริงในระบบนั้น อย่างเช่น “Times New Roman” หรือ “Angsana New” เป็นต้น ภาษา Java เริ่มมีการใช้ face name ตั้งแต่ JDK1.2 เป็นต้นมา นั่นคือเราอาจอ้างถึงชื่อ font ได้โดยใช้ logical name หรือ face name ก็ได้ แต่นับสนุนให้ใช้ face name มากกว่า คลาส Font มี

```
Public String detFontName();
```

```
Public String detFontName(Locale);
```

ซึ่งจะให้ face name ของ font นั้นออกมา

เราสามารถตรวจสอบว่า JDK ที่ใช้นี้มี fonts ทั้ง logical และ face name ไต่บ้างโดยใช้

```
Public abstract Font getAllFonts() [];
```

เป็น method ที่อยู่ในคลาส GraphicsEnvironment ซึ่งเราสามารถเรียกขอได้โดย

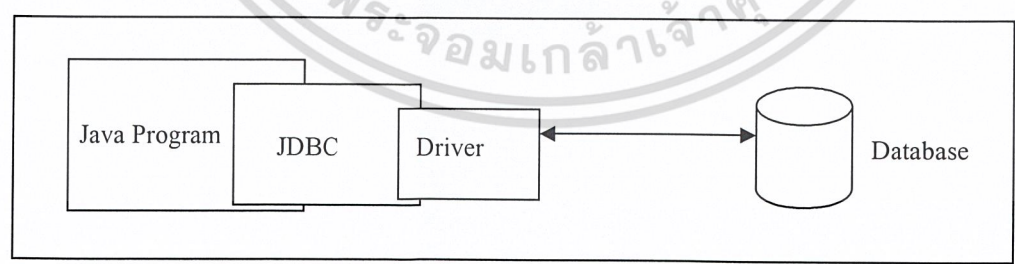
```
Public static synchronized GraphicsEvironment getLocalGraphicsEnvironment();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.10 ระบบจัดการฐานข้อมูล JDBC (Java Data base Connection)

ธุรกิจและองค์กรมีข้อมูลที่ต้องเก็บรักษา และนำออกมาใช้ ระบบฐานข้อมูลจึงเป็นสิ่งจำเป็นในโปรแกรมเชิงธุรกิจ แต่ผู้ผลิตระบบฐานข้อมูลมีมากมายหลายยี่ห้อ มีขนาดและความสามารถแตกต่างกัน วิธีใช้งานก็ต่างกันไปด้วย การเขียนโปรแกรมที่ต้องติดต่อเข้าในระบบฐานข้อมูลจึงยุ่งยากกว่าที่ควรจะเป็น ต่อมาผู้ผลิตระบบฐานข้อมูลส่วนใหญ่ยอมรับ Structured Query Language (SQL) เป็นภาษามาตรฐานสำหรับจัดการกับข้อมูล แต่อย่างไรก็ตาม ระบบฐานข้อมูลแต่ละยี่ห้อ ยังสนับสนุน SQL ต่างกันอยู่บ้าง และก่อนที่จะโปรแกรมจะสามารถส่ง SQL ให้แก่ระบบฐานข้อมูลได้ ก็ต้องติดต่อกับระบบฐานข้อมูลให้ได้ก่อน ซึ่งวิธีติดต่อยังต่างกันไปในช่วงสิบปีที่ผ่านมา มีการพัฒนาในทางที่ดีเกิดขึ้นสองอย่าง ข้อหนึ่ง คือ ผู้ผลิตระบบฐานข้อมูลส่วนใหญ่ตกลงยอมรับ SQL-92 เป็นมาตรฐาน ทำให้เป็นโปรแกรมที่ใช้ SQL-92 สามารถส่งคำสั่งไปทำงานในระบบฐานข้อมูลไปเกือบทุกยี่ห้อ ข้อที่สอง คือ บริษัทผู้ผลิตซอฟต์แวร์หลายแห่งพยายามสร้างมาตรฐานสำหรับให้โปรแกรมติดต่อเข้าใช้งานระบบฐานข้อมูล เพียงแค่บริษัท Microsoft เข้าเดียวก็หลายมาตรฐาน เช่น ODBC, ADODB, DAO และ OLEDB เป็นต้น มาตรฐานเหล่านี้ช่วยให้โปรแกรมสามารถติดต่อเข้าใช้งานระบบฐานข้อมูลยี่ห้อต่างๆ ได้ด้วยวิธีเดียวกัน จึงไม่ต้องสร้างโปรแกรมสำหรับใช้งานระบบฐานข้อมูลแต่ละยี่ห้อ แต่ปัญหาคือมาตรฐานเหล่านี้ยังขึ้นกับภาษาและ platform ที่ใช้งานรวมทั้งบางมาตรฐานไม่ได้ถูกสร้างขึ้นมาสืบสนับสนุนการเรียกใช้จากภาษาเชิงวัตถุ

เมื่อตอนที่ภาษา Java ออกมาใหม่ๆ (ในราวปี 1995) ยังถูกถือว่าเป็นไม่ใช่ภาษาที่ใช้งานได้จริง เพราะยังไม่สามารถ ติดต่อกับระบบฐานข้อมูลได้ ดังนั้นในปี 1996 บริษัท Sun Microsystems Inc. จึงพัฒนา java Database Connection(JDBC) โดยแบ่งออกเป็นสองส่วนคือ JDBC api เป็นชุดของคลาส และ Interfaces เพื่อให้โปรแกรมภาษา Java ใช้ติดต่อกับฐานข้อมูล นำไปสร้าง drivers ที่ติดต่อกับระบบฐานข้อมูลของเขา ทำให้โปรแกรมไม่ผูกติดกับระบบฐานข้อมูลที่ใช้ รวมทั้งไม่จำเป็นต้องทราบรายละเอียดวิธีติดต่อกับระบบฐานข้อมูลนั้น การติดต่อระหว่างโปรแกรมกับระบบฐานข้อมูลแสดงได้ดังในรูปนี้



รูปที่ 2.5 แสดง การติดต่อระหว่างโปรแกรมกับระบบฐานข้อมูล

ผู้ผลิตระบบฐานข้อมูลแต่ละยี่ห้อจะสร้าง drivers สำหรับระบบฐานข้อมูลนั้น โดย drivers จะแปลคำสั่งจาก JDBC เป็นคำสั่งสำหรับเข้าใช้งานระบบฐานข้อมูล ส่วนโปรแกรมภาษา Java จะส่งคำสั่งสำหรับติดต่อกับระบบฐานข้อมูลเป็น SQL ไปที่ JDBC แล้ว JDBC เปลี่ยนคำสั่งนั้นเป็นคำสั่งสำหรับ drivers

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การติดตั้ง drivers สำหรับระบบฐานข้อมูลนั้นอาจทำได้โดยโปรแกรมติดตั้งเอง หรือจะระบบ drivers เมื่อโปรแกรมทำงานก็ได้ ดังนั้นโปรแกรมที่จะใช้ระบบฐานข้อมูลจึงเป็นอิสระจากระบบ ฐานข้อมูล คือเราสามารถสร้างโปรแกรมได้โดยไม่ต้องทราบเลยว่า ระบบฐานข้อมูลที่จะใช้จริง ๆ เป็นยี่ห้อใด และหากเปลี่ยนระบบฐานข้อมูลก็ไม่ต้องเขียนโปรแกรมใหม่ทั้งหมด เพราะ โปรแกรมจะติดต่อออกไปเป็น sql ผ่านทาง JDBC ที่เป็นมาตรฐาน อีกทั้งโปรแกรมหนึ่งอาจมี drivers ติดตั้งอยู่ได้หลายตัว ทำให้โปรแกรมหนึ่งสามารถติดต่อกับระบบฐานข้อมูลได้หลายตัวพร้อมกัน

JDBC รุ่น 1.0 ถูกเพิ่มเข้ามาในภาษา Java ตอนที่ เป็น JDK 1.1 ต่อมามีการพัฒนาเปลี่ยนแปลงอีกสองครั้งใน JDK รุ่นถัดมา แสดงได้ดังนี้

JDBC	JDK
1.0	1.1
2.0	1.2
3.0	1.4

ไม่ว่าจะเป็น JDBC รุ่นใดก็ตาม จะต้องมีการกำหนดเงื่อนไขพื้นฐานสามอย่าง คือ

1. เริ่มต้นติดต่อกับระบบฐานข้อมูล
2. ส่งคำสั่ง sql ไประบบฐานข้อมูล
3. รับผลลัพธ์จากระบบฐานข้อมูลเข้ามาจัดการ

JDBC ทั้งสามรุ่นสนับสนุนกลไกสำหรับใช้งานขึ้นพื้นฐานเหมือนกัน แต่ JDBC รุ่นหลังจะเพิ่มกลไกบางอย่าง ที่รุ่นเก่านั้นไม่มี นับเริ่มตั้งแต่ JDBC 2.0 เป็นต้นมา คลาสและ interfaces ของ JDBC ถูกแบ่งออกเป็นสอง packages หนึ่งคือ “java.sql” มีกลไกขั้นพื้นฐาน ซึ่งเป็นข้อกำหนดว่า drivers ของระบบฐานข้อมูลยี่ห้อใดที่จะใช้งานกับ โปรแกรมภาษา Java ต้องสนับสนุน “java.sql” ทั้งหมด สองคือ “javax.sql” ซึ่งเรียกว่า JDBC optional package มีกลไกที่ drivers ของระบบฐานข้อมูลที่ระบบ J2ee ต้องสนับสนุน แต่ระบบฐานข้อมูลทั่วไปอาจจะไม่สนับสนุนก็ได้

2.10.1 JDBC Drivers

เมื่อโปรแกรมภาษา Java ต้องการติดต่อกับระบบฐานข้อมูลใด ก็จะต้องมี drivers สำหรับระบบฐานข้อมูลนั้น เพื่อทำหน้าที่เชื่อมต่อระหว่างโปรแกรมกับข้อมูล แต่ระบบฐานข้อมูลมีความแตกต่างกันไปในแต่ละยี่ห้อ ดังนั้น drivers ที่สร้างขึ้นสำหรับแต่ละ JDBC จึงถูกสร้างขึ้นด้วยวิธีที่ต่างกันไปซึ่งส่งผลถึงสภาพแวดล้อมไรตอนใช้งานด้วยเราสามารถแบ่ง drivers ตามวิธีที่ถูกสร้างขึ้นมาได้ 4 ประเภท ดังนี้

Type 1.JDBC-ODBC driver เป็นประเภทที่ถูกสร้างขึ้นโดยใช้ native code เพื่อทำหน้าที่เป็นตัวกลางระหว่างโปรแกรมกับ ODBC drivers ข้อดีของการใช้ bridge drivers คือ ODBC drivers สามารถติดต่อกับระบบฐานข้อมูลได้หลายยี่ห้อ โดยเฉพาะที่ทำงานได้ใน Microsoft Windows แต่ข้อเสียคือ มีการเปลืองคำสั่งเพิ่มขึ้นอีกระดับหนึ่งทำให้ทำงานช้าเมื่อเปรียบเทียบกับ drivers ประเภทอื่น ๆ และที่เครื่องด้าน clients ต้องมีการติดตั้งโปรแกรมของ ODBC รวมทั้งต้องทำ configuration ให้มองเห็น databases ที่จะใช้ ตัวอย่าง drivers ประเภทนี้คือ JDBC-ODBC drivers ซึ่งอยู่ใน package “un.jdbc.odbc.JdbcOdbcDriver” ที่มากับ JDK นับตั้งแต่ 1.1 เป็นต้นมา ที่จริงแล้ว จุดประสงค์ของการสร้าง JDBC-ODBC drivers คือ เพื่อให้ผู้เขียนโปรแกรมทดลองเปลี่ยนจาก ODBC เป็น JDBC แต่ในปัจจุบัน(กลางปี 2545) ระบบฐานข้อมูลเกือบทุกยี่ห้อ มี drivers ประเภทอื่นที่จะกล่าวถึงข้างล่างนี้ ซึ่งทำงานได้เร็วกว่า และเชื่อถือได้มากกว่า ดังนั้นจึงแนะนำว่า ควรใช้ JDBC-ODBC drivers ในการทดลองหรือเรียนรู้เท่านั้น ส่วนในการใช้งานจริงควรใช้ drivers ประเภทอื่น

Type 2.Native API partly java driver เป็นประเภทที่ถูกสร้างขึ้นโดยใช้ native code แล้วนำมาสร้างเป็นคลาสของภาษา Java ในการใช้งานจะต้องติดตั้งทั้ง native code และคลาสของ Java ที่เครื่องด้าน clients ทุกตัวที่ใช้งาน เช่นเดียวกับแบบแรก แต่เนื่องจากไม่ต้องผ่าน drivers ตัวกลางอย่าง ODBC และมีส่วนที่ทำงานด้วย native code จึงทำงานได้เร็วกว่าแบบอื่น ๆ ยกเว้นแบบที่ 4 ตัวอย่าง driver ประเภทนี้ได้แก่ Oracle Call Interface(OCI) driver และยังมีของบริษัท Sybase, Informix และ IBM ด้วย

Type 3.JDBC-Net pure Java driver เป็นประเภทที่สร้างด้วยภาษา Java ทั้งหมด โดย driver จะแปลคำสั่งจาก JDBC เป็นคำสั่งที่ใช้ส่งผ่าน network ไปที่ server ของระบบฐานข้อมูล จากนั้น server จะแปลคำสั่งที่ได้รับเป็นคำสั่งสำหรับเชื่อมต่อกับฐานข้อมูล ข้อดีคือ drivers ชนิดนี้สามารถติดต่อกับฐานข้อมูลที่ยี่ห้อใดก็ได้ หาก servers ของระบบฐานข้อมูลนั้นใช้ network protocol เดียวกัน และเนื่องจาก drivers ประเภทนี้ถูกสร้างขึ้นด้วยภาษา Java ทั้งหมด จึงสามารถถูกดาวน์โหลดไปที่เครื่องด้าน clients โดยอัตโนมัติ แต่ก็มีข้อเสียคือ เพิ่มความยุ่งยากในการจัดการด้านความปลอดภัย ตัวอย่างของ drivers ประเภทนี้ได้แก่ Web logic ของ Tengah

Type 4.Native protocol pure Java driver เป็นประเภทที่สร้างด้วยภาษา Java ทั้งหมด เหมือนกับแบบที่ 3 แต่ drivers จะทำหน้าที่ติดต่อกับระบบฐานข้อมูลโดยตรง ทำให้มีประสิทธิภาพการทำงานมาก เรามักเรียก drivers ประเภทนี้ว่า thin drivers ระบบฐานข้อมูลที่มี drivers ประเภทนี้ได้แก่ Oracle, Sybase, Informix, IBM DB2, INprise Interbase และ Microsoft SQL Server

ข้อเสนอแนะในการเลือกใช้งาน drivers ให้เหมาะสมกับงาน มีดังนี้

1. ถ้าระบบของเราใช้ระบบฐานข้อมูลหนึ่งอยู่แล้ว เราต้องใช้ driver ที่มีมากับระบบฐานข้อมูลนั้น ซึ่งโดยปกติเกือบทุกยี่ห้อให้ฟรี และอาจมีรุ่นปรับปรุงใหม่ให้ดาวน์โหลดได้เรื่อย ๆ หากไม่จำเป็นแล้วไม่ควรใช้ driver ที่ผลิตด้วยบริษัทที่ไม่ใช่ผู้ผลิตระบบฐานข้อมูลนั้น
2. การเลือกประเภทของ drivers มีผลเฉพาะในตอนใช้งานโปรแกรม แต่ในระหว่างพัฒนาโปรแกรมเราจะใช้ drivers ประเภทใดก็ได้ หากในตอนใช้งานเราต้องการความเร็วควรเลือก drivers ประเภทที่เป็น native API หรือ native protocol หากเราไม่สามารถติดตั้งโปรแกรมที่เครื่องของ clients ก็ควรเลือก drivers ประเภทนี้เป็น pure Java เพื่อให้สามารถดาวน์โหลดไปที่เครื่องของ clients ได้ง่าย

2.10.2 Establishing a Connection

ระบบฐานข้อมูลโดยทั่วไปมีขั้นตอนการสร้างการติดต่อ รวมทั้งการส่งคำสั่ง SQL ออกไปและรับผลลัพธ์กลับมาเหมือนกัน เพียงมีชื่อ drivers และวิธีการอ้างถึงชื่อ databases ที่แตกต่างกันไปในแต่ละยี่ห้อ ดังนั้นในบทนี้เราเริ่มต้นโดยแสดงการเชื่อมต่อกับระบบฐานข้อมูลสี่ยี่ห้อคือ Microsoft Access, Oracle, MySQL และ Cloudscape เมื่อเชื่อมต่อกับระบบฐานข้อมูลได้แล้ว จะแสดงรายละเอียดการติดต่อกับระบบฐานข้อมูล Cloudscape เพียงยี่ห้อเดียว เพราะสำหรับระบบฐานข้อมูลยี่ห้ออื่นก็จะคล้ายกัน

ที่เลือกแสดงการติดต่อกับ Microsoft Access, MySQL และ Oracle เพราะเป็นระบบฐานข้อมูลที่ได้ความนิยม ใช้อย่างแพร่หลาย แต่ภายหลังเราใช้ Cloudscape เพียงอย่างเดียว เพราะสะดวกในการติดตั้ง

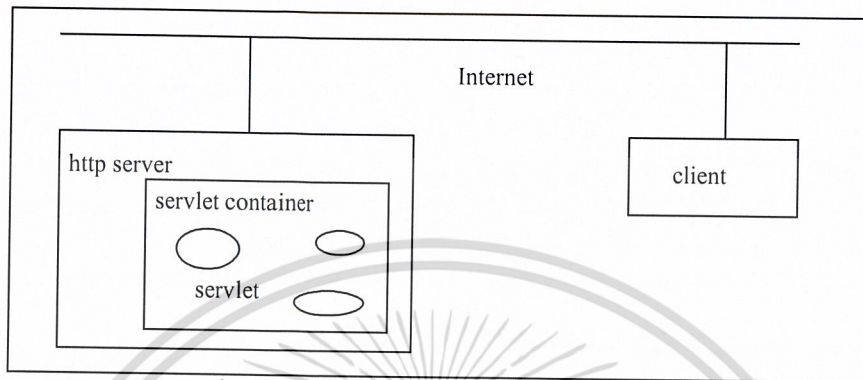
เราจะมีโปรแกรมภาษา Java สำหรับแสดงการติดต่อกับระบบฐานข้อมูลแต่ละยี่ห้อ โดยสมมติว่า จะมี table หนึ่งชื่อ STUDENT ถูกสร้างไว้แล้ว หรือสร้างโดยใช้โปรแกรมภาษา Java หรือคำสั่งของระบบฐานข้อมูลนั้นกำหนดว่ามีโครงสร้างดังนี้

Name	type
id	INTEGER
name	VAR CHAR(30)
department	VAR CHAR(15)
gpa	REAL

และให้มีข้อมูลเริ่มต้นจำนวนหนึ่ง สำหรับให้โปรแกรมทดสอบนำข้อมูลใน STUDENT table ออกมาพิมพ์ทาง standard output แต่ในตอนนี้เป็นเตรียมการเพื่อเชื่อมต่อกับระบบฐานข้อมูลเท่านั้น ดังนั้นเราจะแสดงโปรแกรมทดสอบเพื่อใช้งานไปก่อน แล้วจะอธิบายรายละเอียดของโปรแกรมในภายหลัง

2.11 Servlets

Servlets คือรูปแบบของโปรแกรมภาษา Java ที่ถูกสร้างขึ้นเพื่อเป็น server side programs โดยข้อกำหนดแล้ว Servlets ต้องทำงานอยู่ในสถานะแวดล้อมพิเศษชนิดหนึ่งที่เรียกว่า Servlet container ซึ่งจะถูกสร้างขึ้น http Servlets เพื่อทำงาน Servlets ดังรูปต่อไปนี้



รูปที่ 2.6 แสดงสถานะแวดล้อมของ servlet

Servlets อาจถูกเรียกทำงานจากโปรแกรม client ที่มักจะเป็น browser หรือเป็นโปรแกรมใด ๆ เช่น applets หรือ java applications ก็ได้ที่สามารถติดต่อผ่านระบบเครือข่ายโดย http protocol มาที่ http server ที่ให้บริการ Servlets นั้น สังเกตว่าหาก client เป็น browser มักจะติดต่อกับ Servlets โดยใช้ http request/response ได้เลยซึ่งจะสะดวกมาก แต่ถ้า client เป็นโปรแกรม Java ใด ๆ มักจะติดต่อกับ Servlets โดยใช้ URLConnection และส่งข้อมูลผ่านทาง input/output streams

โปรแกรมที่จะถูกทำงานภายใน Servlets containers ได้ ต้องเป็นคลาสที่ implements interface ที่ชื่อว่า Servlet ซึ่งอยู่ใน javax.servlet package มีรายละเอียดดังนี้

Public interface Servlet{

Public abstract void init (ServletConfig) throws ServletException;

Public abstract ServletConfig getServletConfig();

Public abstract void service(ServletRequest, ServletResponse)

throws ServletException, IOException;

Public abstract String getServletInfo();

Public abstract void destroy();

}

การสร้าง servlets เป็นการเขียนโปรแกรมแบบ event-driven programming นั่นคือ servlets ไม่มี threads ของตัวเอง คลาสของ servlets ต้อง implements methods เหล่านี้ไว้เพื่อให้ threads ของ servlet containers เรียกทำงาน เนื่องจาก servlets containers เป็นผู้ควบคุมวัฏจักรชีวิตและการทำงานของ servlets เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อ servlet container โหลดคลาสของ servlet เข้ามา ก็จะสร้าง instance ของคลาสนั้นขึ้นทันที แล้วเรียกทำงาน `Init()` โดยส่งข้อมูลเกี่ยวกับ servlet นี้เข้าไปเป็นพารามิเตอร์ `servletConfig` เพื่อทำการกำหนดค่าเริ่มต้นก่อนที่ servlet จะเริ่มให้บริการ หาก `init()` ไม่สามารถให้จบอย่างปกติได้จะเกิด `servletException` ขึ้น และมีข้อกำหนดว่า servlet container ต้องรับประกันว่า `init()` ของ servlet หนึ่งจะถูกเรียกทำงานได้เพียงหนึ่งครั้งเท่านั้น รวมทั้ง `init()` ต้องทำงานเสร็จก่อนจะมี request มาที่ servlet นั้น

หลังจาก `init()` ถูกทำงานแล้ว เมื่อมี request มาที่ servlet นั้น servlet container ก็จะเรียก `service()` เพื่อทำงานบริการของ servlet นั้น `service()` มีพารามิเตอร์สองตัวคือ `servletRequest` สำหรับให้ servlet อ้างถึง http request ที่มาจาก client และ `servletResponse` สำหรับให้ servlet ส่งผลลัพธ์กลับไปเป็น http response ให้แก่ client

Servlet containers เรียก `destroy()` เมื่อจะทำลาย instance ของ servlet นั้น เช่น เมื่อ http server จะเลิกทำงาน มีข้อกำหนดว่า servlet containers ต้องรอให้ threads ที่ทำงาน `service()` ของ servlet นั้นเลิกทำงานให้หมดก่อน แล้วจึงเรียก `destroy()` ได้ และรับประกันว่าระหว่างที่ `destroy()` ทำงาน รวมทั้งหลังจากทำงานเสร็จแล้วยังมี requests มาที่ servlet นั้นอีก ก็จะไม่ทาน `service()` ของ servlet นั้น

โดยปกติ servlet มักจะเก็บ `servlet config` ที่เป็นพารามิเตอร์ของ `init()` ไว้เป็น data member ของคลาสเพื่อใช้อ้างถึงข้อมูลที่อยู่ใน `servlet config` นั้น แต่อาจเป็นไปได้ที่โปรแกรมอื่น ๆ หรือ servlet อื่นต้องการทราบ `servlet config` นั้นเช่นกัน ดังนั้นเราจึงควรสร้าง `getServletConfig()` ให้ส่งค่า `servlet config` นั้นออกมา แต่หากเราไม่ต้องการ servlet ของเราเปิดเผย `servlet config` แก่ผู้อื่น ก็ให้ค่าออกมาเป็น `null`

ส่วน `getServletInfo()` จะส่งค่าออกมาเป็น string ที่เป็นข้อมูลเกี่ยวกับ servlet นั้น ซึ่งเราอาจกำหนดเป็นอะไรก็ได้ ขึ้นกับที่เราจะใช้ประโยชน์ข้อมูลนี้ในด้านใด

2.11.1 Compiling a Srevlet Class

คลาสและอินเตอร์เฟซที่เกี่ยวกับ servlets ถูกสร้างไว้ใน javax. Servlet package แต่ว่า package นี้ไม่ได้ถูกรวมมากับ j2sdk จนถึงปัจจุบัน (กลางปี 2546) ที่เป็นรุ่น 1.4.1_01 ดังนั้นเราต้องทำการติดตั้ง package นี้เพื่อใช้งานเองโดยเริ่มจากตรวจสอบที่

<http://java.son.com/products/servlet/download.html>

จะเห็นว่าเราอาจได้ javax. Servlet package มาจากสองแหล่งคือ j2sdkee และ Jakarta Tomcat ปัจจุบัน servlets ถูกพัฒนามาถึงรุ่น 2.4 และ j2sdkee สนับสนุน servlet รุ่นต่างกันดังนี้

J2EE	Servlet
J2sdkee 1.3.1	2.3
J2sdkee 1.4	2.4

ส่วน Jakarta Tomcat สนับสนุน servlet รุ่นต่างกันดังนี้

Jakarta Tomcat	Servlet
Tomcat 3.3	2.2
Tomcat 4.1.18	2.3
Tomcat 5.x	2.4

หากเรามี j2sdkee ติดตั้งอยู่แล้วให้ตรวจสอบในไฟล์ %J2EE_HOME%\lib\j2ee.jar จะพบว่า มี javax. Servlet package อยู่แล้ว ดังนั้น หากเรามี CLASSPATH ชี้ไปที่ไฟล์ j2ee.jar นี้ก็จะสามารถใช้งาน คลาสเกี่ยวกับ servlet ได้ ทดสอบโดย

สมมุติเรามี J2EE_HOME ชี้ที่ได้เรกทอรีที่ติดตั้ง j2sdkee เช่น c:\jadkee1.3.1 อยู่แล้ว เรากำหนด CLASSPATH ได้โดย

Set CLASSPATH=%J2EE_HOME%\lib\j2ee.jar;

แล้วทำการคอมพไล์ MyServlet.java โดย

Javac MyServlet.java

ขอแนะนำว่าไม่ควรนำ j2ee.jar มาไว้ที่ %JAVA_HOME%\jre\lib\ext เพราะไฟล์ j2ee.jar มี ขนาดใหญ่มาก จะทำให้ class loader ใช้เวลามากค้นหาคลาส

หากเราไม่ได้ใช้งาน j2ee ก็ใช้ Jakarta Tomcat ซึ่งมีไฟล์ jar ที่มี javax.servlet และมีขนาดเล็กกว่า มาก เมื่อตอนแรกบริษัท Sun MicroSystem เป็นผู้เริ่มพัฒนา serlets ขึ้นเป็นส่วนหนึ่งของภาษา Java เมื่อ พัฒนาถึงรุ่น 2.0 ก็มอบหมายให้ Apache Oragnization เป็นผู้พัฒนาต่อและกำหนดมาตรฐานของ servlets โครงการนี้เรียกว่า Jakarta Tomcat (หรือเรียกสั้น ๆ ว่า Tomcat) ทำการพัฒนาคลาสและอินเตอร์เฟซ เกี่ยวกับ servlets เพื่อแจกจ่าย Open source software พร้อมทั้งกำหนดมาตรฐานของ servlet containers โดยสร้าง servlet container เพื่อเป็นแม่แบบสำหรับ servlet container ที่สนับสนุน servlet รุ่นนั้น ๆ เรา สามารถดาวน์โหลดไฟล์สำหรับติดตั้ง tomcat ได้จาก <http://jakart.apache.org>

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ไฟล์ติดตั้ง Tomcat มีทั้งสองแบบ คือ standard และ JDK11.4 LE ซึ่งมาขนาดเล็กกว่าแบบ standard เพราะตัด packages เกี่ยวกับ JavaMail, Java Activation Framework, Xerces (XML parser), JNDI และ JDBC ออกไปเหมาะสำหรับใช้ศึกษา servlets ขึ้นเบื้องต้นที่ไม่ใช้กลไกเหล่านี้ แต่หากจะพัฒนา servlets เพื่อใช้งานจริง ๆ ควรเลือกใช้แบบ standard

ไฟล์ติดตั้ง Tomcat ยังแบ่งได้เป็นอีกสองแบบ คือแบบที่เป็นไฟล์ .exe มีโปรแกรม wizard ช่วยทำการติดตั้งโปรแกรมพร้อมทั้งกำหนดค่า environment variables และ startup menu ให้โปรแกรมติดตั้งแบบนี้ใช้งานง่าย แต่ยุ่งยากในการ uninstall ส่วนอีกแบบเป็นไฟล์ .zip เราต้องทำการติดตั้งโดยการ unzip เอง แต่สะดวกในตอน uninstall (คือเพียงแค่ลบไดเรกทอรีของ Tomcat เท่านั้น และยังสามารถติดตั้ง Tomcat หลายรุ่นบนเครื่องเดียวกันได้ เราใช้ Tomcat เพื่อศึกษาหรือพัฒนา sevlts ควรเลือกใช้ไฟล์ติดตั้งแบบ.zip เนื่องจาก tomcat มีการพัฒนารุ่นใหม่ออกมาอยู่เสมอ จึงมักต้อง uninstall รุ่นเก่า และติดตั้งรุ่นใหม่อยู่เรื่อย ๆ

Tomcat ผ่านการเปลี่ยนแปลงมาแล้วหลายรุ่น รุ่นที่สำคัญในระยะหลังมีดังนี้

Tomcat 3.1 ปัจจุบันไม่สนับสนุนให้ใช้แล้ว แต่ยังมีใช้อยู่ระบบที่ทำงานมานานแล้ว

Tomcat 3.2 เริ่มสนับสนุน servlet 2.2 และ jsp 1.1 มีการทำงานที่มั่นคงมาก ปัจจุบันยังคงมีใช้อยู่ระบบหลายระบบ

Tomcat 3.3 ปรับปรุงด้านประสิทธิภาพจากรุ่น 3.2

Tomcat 4.x มีชื่อรุ่นว่า Catalina สร้างขึ้นใหม่ทั้งหมดเพื่อสนับสนุน servlet 2.3 และ jsp 1.2

เราต้องเข้าใจโครงสร้างไดเรกทอรีของ Tomcat จึงสามารถทำการ config และ deploy ได้อย่างถูกต้อง ดังนั้น จึงขออธิบายหน้าที่ของไดเรกทอรีต่าง ๆ ดังนี้

- Bin .bat สำหรับเริ่มและหยุดทำงาน Tomcat
- Common มีคลาสและไฟล์ jar ที่ sever และ web applications ใช้ร่วมกัน
- Conf มีไฟล์ xml สำหรับ configuration ไฟล์ properties และ policy สำหรับ server
- Logs เป็นที่เก็บไฟล์ log ที่ server เขียนออกมาขณะทำงาน
- Server มีคลาสและไฟล์ jar ที่สำหรับ server ใช้งาน
- Shared เป็นที่เก็บคลาสและไฟล์ jar ที่ web applications สามารถนำมาติดตั้งเพื่อใช้งานร่วมกัน
- Webapps เป็นที่สำหรับติดตั้ง web applications
- Work เป็นที่เก็บไฟล์หรือไดเรกทอรีที่ถูกสร้างเพื่อใช้ชั่วคราว
- Temp เป็นที่เก็บไฟล์ชั่วคราวของ JVM(java.io.tmpdir)

หากเราเพิ่งติดตั้ง Tomcat เสร็จ จะยังไม่มีไฟล์ในไดเรกทอรี logs, work และ shared จนเมื่อเราทำงาน Tomcat เป็นครั้งแรกจึงจะมีไฟล์ในไดเรกทอรี logs และ work ส่วนไดเรกทอรี shared เราจะต้องนำคลาสหรือไฟล์ jar ไปติดตั้งเอง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.11.2 Tomcat Http Server and Servlet Container

โดยปกติ servlets จะต้องทำงานอยู่ภายใน servlet containers แต่เรามักเรียกให้ servlets ทำงานผ่านทาง http servers ดังนั้น servlets containers จึงต้องทำงานร่วมกับ http servers โดยที่ http servers จะรับ requests มาจาก client ก่อน หากตรวจสอบแล้วพบว่าเป็นการ request มาที่ servlets จึงส่ง request นั้นมาที่ servlet container ให้ทำงาน servlet นั้น

โดยทั่วไป servlet containers อาจ แบ่งได้เป็น 2 ประเภท ตามความสัมพันธ์กับ http servers คือ

1. Standalone Servlet Container เป็น servlet container ติดมากับ http server นั้นเลย ตัวอย่างคือ Java Web Server และ Apache JServ เป็นต้น ข้อดี คือติดตั้งง่าย และทำงานได้อย่างมีประสิทธิภาพ เพราะว่า http server แบบนี้ถูกออกแบบมาให้ทำงานสัมพันธ์กับ servlet container นั้น โดยตรง แต่ข้อเสียคือ ยากที่จะเปลี่ยนแปลงให้สนับสนุน servlet รุ่นใหม่ ๆ ที่ออกมาภายหลัง คือต้องรอให้ http server นั้นปรับปรุงเป็นรุ่นใหม่จึงจะสามารถสนับสนุน servlet รุ่นใหม่

2. Add-on Servlet Container เป็น โปรแกรม servlet container ที่เราสามารถ plug-in กับ http servers ที่ไม่มี servlet container โดยตรง ตัวอย่างคือ Jrun และ Tomcat servlet container ข้อดี สามารถเปลี่ยนแปลงเฉพาะ servlet container นั้นให้สนับสนุน servlet รุ่นใหม่ ๆ ที่ออกมาที่หลังได้ แต่อาจมีข้อยุ่งยากในการติดตั้งให้ทำงานร่วมกับ http servers

2.11.3 Server Configuration

ในไดเรกทอรี /conf ของ Tomcat มีไฟล์ server.xml สำหรับกำหนด configuration ของ Tomcat server ไฟล์นี้เป็น xml และมีการคอมเมนต์ไว้ค่อนข้างดี ขอแนะนำให้ลองพยายามอ่าน ทำความรู้จักอย่างคร่าว ๆ ก่อนจะเห็นการ config ของ Tomcat แบ่งระดับของส่วนประกอบเป็น

Server หมายถึงหลาย ๆ server ที่ให้บริการในหนึ่ง JVM โดยมี Server tag สำหรับกำหนด port เพื่อรับคำสั่งเลิกทำงาน (shutdown) Tomcat server โดย default ไว้ดังนี้

```
<Server port="8003" shutdown="SHUTDOWN" debug="0">
```

นั่นคือเราสามารถต่อ socket เข้าไปที่ port 8005 และส่งข้อความ "SHUTDOWN" ซึ่งสมมุติว่าใช้เป็น password เข้าไปใน stream เพื่อเลิกทำงาน Tomcat server

Service หมายถึง กลุ่มของหลาย ๆ connector ภายใต้นี้ container (หรือเราเรียกว่าอยู่ภายใน context เดียวกัน ใน server.xml มี service tab ที่ถูกกำหนดไว้เป็น default ดังนี้

```
<Service name="Tomcat-Standalone">
```

ซึ่งระบุไว้ให้บริการแบบ standalone คือมี http server ของตัวเอง หากตรวจสอบดูที่ท้าย ๆ ไฟล์ server.xml นี้จะพบว่ามี

```
<Service name="Tomcat - Apache">
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ที่ถูกคอมเมนต์ไว้สำหรับใช้เมื่อจะทำงานร่วมกับ Apache http server

Connector เป็น endpoint ที่ใช้รับ requests ตอบ responses ในหนึ่ง connector เราอาจกำหนดค่า Values (filters) ก็ได้ ใน server.xml มี Connector tag กำหนด http connector (ชื่อ Coyote Connector) ให้บริการที่ port 8080 ดังนี้

```
<Connector className="org.apache.coyote.tomcat4.CoyoteConnector"
    Port="8080" minProcessors="5" maxProcessors="75"
    enableLookups="true" redirectPort="8443"
    acceptCount="100" debug="0" connectionTimeout="20000"
    useURValidationHack="false" disableUploadTimeout="true" />
```

ค่าพารามิเตอร์มีความหมายดังนี้

- Port กำหนดเบอร์ port ที่ให้บริการ หากเราต้องการเปลี่ยนเป็นเบอร์ 80
- minProcessors กำหนดจำนวน threads สำหรับให้บริการที่จะถูกสร้างขึ้นเมื่อ server เริ่มทำงาน
- maxProcessors กำหนดจำนวน threads มากสุด ที่จะถูกสร้างขึ้นเพื่อให้บริการ หรือกล่าวได้ว่ากำหนดจำนวน requests มากสุดที่จะให้บริการได้พร้อมกัน
- acceptCount กำหนดความยาวของ connection request queue หาก ๆ นี้เต็ม requests ที่เข้ามาในระหว่างนั้นจะไม่ได้รับบริการ
- connection Timeout กำหนดเวลา (เป็น msec) ที่ requests จะรออยู่ใน connection request queue หากจะให้รอโดยจำกัดเวลาให้กำหนดค่าเป็น -1

ในแต่ละ Service เราสามารถกำหนด Engine (หรือเรียกว่า Catalina Entry Point) ซึ่ง requests ที่เข้ามาใน connector ทั้งหมดของ Service นั้นจะถูกส่งไปที่ servlet engine นี้ เพื่อถูกจัดการโดย Valves ที่กำหนดไว้ ตัวอย่างเช่น Engine ของ Tomcat stand alone Service จะตรวจสอบ http request headers แล้วจึงส่ง requests นั้นไปที่ virtual host สำหรับจัดการกับ requests นั้น

ใน server.xml มี Engine tag สำหรับ Tomcat stand alone Service ดังนี้

```
<Engine name="Standalone" defaultHost="localhost" debug="0" />
```

ในแต่ละ Engine เราสามารถกำหนด global components ที่จะถูกใช้โดย Valves ของ Engine นั้น แต่หากมีบาง Valves ต้องการ override components เหล่านั้นก็ได้ ตัวอย่างเช่น Engine อาจกำหนดไฟล์ logger เป็น global component หนึ่ง โดยมี Logger tag ดังนี้

```
<Logger className="org.apache.catalina.logger.FileLogger"
    Prefix=Catalina_og." Suffix = ".txt"
    Timestamp="true"/>
```

Global components อื่น ๆ ของ Engine อาจเป็น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- Context กำหนด ServletContext
- Loader กำหนด class loader เพื่อโหลดคลาสและทรัพยากรในแง่ web applications
- Manager กำหนด class manager ที่สร้างและควบคุม http sessions
- Realm กำหนดข้อมูลของกลุ่ม users

Value คือโปรแกรมที่ทำหน้าที่ filter ที่สามารถแทรกเข้าไปใน requests และ responses เพื่อจัดการกับข้อมูลที่เคลื่อนที่ผ่าน ตัวอย่างเช่น AccessLogValue ซึ่งทำการ log การขอใช้ทรัพยากร ถูกกำหนดไว้โดย Valve tag ดังนี้

```
<Valve className="org.apache.catalina.valves.AccessLogValve"
```

```
Directory="logs"
```

```
Prefix="localhost_access_log." Suffix=".txt"
```

```
Pattern="common" resolveHosts="false"/>
```

ซึ่งจะเริ่ม log การใช้ทรัพยากรลงในไฟล์ที่มีชื่อ localhost_access_log และต่อด้วยตัวเลข ซึ่งไฟล์นี้จะอยู่ในไดเรกทอรี logs ของ Tomcat

ยังมี tag อื่น ๆ อีกหลายตัว แต่เราไม่สามารถอธิบายในที่นี้ได้ทั้งหมด ดังนั้นเมื่อกล่าวถึงเรื่องที่เกี่ยวข้องกับ tag บางตัวเราจึงจะอ้างอิงอีก

2.11.4 Web Applications

แต่ก่อนการสร้าง servlets และ html เพื่อให้บริการแบบ web based จะต้องขึ้นอยู่กับ http server และ servlet container ที่จะใช้นั้นมีโครงสร้างไดเรกทอรีและวิธีการนำโปรแกรมและ documents ขึ้นให้บริการ(deployment) อย่างไร ทำให้โปรแกรม web based ที่ถูกพัฒนาจนใช้งานได้บน http server หนึ่งแล้วเมื่อนำไปใช้อีกบน http servers ต่างยี่ห้อ หรือมีวิธีการ deploy ต่างกัน ต้องถูกปรับปรุงเปลี่ยนแปลงอย่างมาก เสียทั้งเวลา และค่าใช้จ่าย ดังนั้นเพื่อให้การนำโปรแกรม web based มาใช้อีกง่ายขึ้น ในข้อกำหนดของ servlet2.2 จึงแนะนำ Web Applications (หรือเรียกสั้น ๆ ว่า web apps) ขึ้นโดยใช้ Tomcat3.2 เป็นแม่แบบของ http server และ servlet container ที่สามารถให้บริการ web apps

เราให้คำจำกัดความว่า หนึ่ง web app คือ กลุ่มของโปรแกรม (ซึ่งอาจจะเป็น servlets, applets, java classes หรือ beans) และทรัพยากร (ซึ่งอาจจะเป็นไฟล์ html, jsp, xml, txt, gif, jpeg, au, wav หรืออื่น ๆ) ที่สามารถทำงานร่วมกัน เพื่อให้บริการบางอย่าง แบบ web based ได้ โดยมีข้อกำหนดว่าหนึ่ง web app จะต้องถูกสร้างไว้ในหนึ่ง ไฟล์ war ซึ่งก็คือ ไฟล์ที่ถูกบีบอัดด้วยวิธี pkzip เหมือนกับไฟล์ zip เพียงแต่มีโครงสร้างไดเรกทอรีเป็นข้อกำหนดพิเศษสำหรับ web apps จึงเรียกไฟล์ war นั้นเอง ด้วยวิธีนี้ การติดตั้ง web app หนึ่งจะทำได้โดยก๊อปปี้ไฟล์ war หนึ่งไปวางให้ถูกไดเรกทอรีเท่านั้น

Servlet containers ที่สนับสนุนข้อกำหนด servlet2.2 ขึ้นไป ต้องมีไดเรกทอรีหนึ่งสำหรับให้เราวางไฟล์ war ของ web apps ที่จะถูก deployed บน server นั้น และต้องสามารถขยายไฟล์ war เหล่านั้นออกมาเป็น web apps และทำการ deploy โดยอัตโนมัติได้ ตัวอย่างเช่น Tomcat มีไฟล์ server.xml (ในไดเรกทอรี conf) ที่มี Host tag ดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```
<Host name="localhost" debug="0" appBase="webapps"
```

```
unpackWARs="true" autoDeploy="true">
```

ซึ่งกำหนด virtual host ชื่อ localhost บนเครื่องที่ทำงาน โดยมี appBase ระบุไดเรกทอรีสำหรับติดตั้ง web apps อยู่ที่ %CATALINA_HOME%/webapps พร้อมกับระบุพารามิเตอร์ unpackWARs และ autoDeploy เป็น true ซึ่งหมายถึงให้ทำการขยายไฟล์ war ออกมาเป็น web apps และทำการติดตั้งให้อย่างอัตโนมัติ ดังนั้นหากเราก็อปปี้ไฟล์ war ไปไว้ที่ไดเรกทอรีนี้ web app ในไฟล์ war นี้จะถูกติดตั้งโดยอัตโนมัติ

เมื่อตอนที่ Tomcat เริ่มทำงานทุกครั้งจะตรวจสอบว่าในไดเรกทอรี webapps มีไฟล์ war ที่ยังไม่ถูกขยายหรือไม่ ถ้ามีก็จะขยายออกมาสร้างเป็นไดเรกทอรีชื่อเหมือนกับไฟล์ war นั้น ในไดเรกทอรี webapps นี้ พร้อมกับทำการอ่านไฟล์ configuration ของ web app นั้นแล้วติดตั้งกับ server ให้โดยอัตโนมัติ สังเกตว่าหลังจากไฟล์ war ถูกขยายออกมาสร้างเป็นไดเรกทอรีแล้ว ในการเริ่มทำงาน Tomcat ครั้งต่อ ๆ ไปจะไม่สนใจไฟล์ war นั้นอีก (เพื่อไม่ต้องเสียเวลา startup นาน) ดังนั้นถึงแม้เราลบไฟล์ war นั้นออกไป web app ของมันที่ถูก deploy ไว้แล้วก็ยังทำงานได้ แต่ถ้าเราลบไดเรกทอรีของ web app นั้นออกไปไฟล์ war ของมันจะถูกขยายและติดตั้งอีก

2.11.5 WAR Files

หากเราจะนำ myApp web app ที่พัฒนาเสร็จแล้วไป deploy บน http server เครื่องอื่น ที่จริงแล้วเราสามารถก๊อปปี้ไดเรกทอรี /myApp และที่อยู่ภายในทั้งหมดไปวางไว้ใน /webapps ของเครื่องที่จะติดตั้งนั้นก็ได้อีก แต่เราทำได้ดีกว่านั้น โดยทำการบีบอัดไฟล์ทั้งหมดในไดเรกทอรี /myApp เป็นไฟล์ .war ซึ่งเป็นไฟล์ผลลัพธ์ของการบีบอัดข้อมูลเหมือนกับไฟล์ .jar นั่นเอง และทำได้โดยใช้ jar.exe เช่นกัน เมื่อได้เป็นไฟล์ .war แล้วจะสะดวกทั้งในการนำไปใช้อีกและการนำไปติดตั้ง

2.11.6 Servlet Aliases

แม้มีความจำเป็นว่า servlets ต้องมีชื่อสำหรับให้ servlet container อ้างอิง (เพื่อจะทำงานบางอย่างได้ เช่น การขอ request dispatcher servlet ซึ่งจะกลางถึงภายหลัง) แต่หากเราต้องทำการ config ชื่อ servlets คือกำหนด servlets-name, servlets-class และ url-pattern ในไฟล์ web.xml ให้แก่ทุก ๆ servlets ก็อาจยุ่งยากไปเกินไป ดังนั้นถ้าต้องการเพียงแค่เรียกใช้บริการ servlets ได้ เราก็อาจหลีกเลี่ยงการ config ชื่อ servlets โดยการทำให้ servlets aliases คือมีตัวอย่างเช่น servlet นำหน้าชื่อ servlet ใน servlet url จากนั้นทำการ configuration ให้ servlet container ตรวจสอบหาค่า ๆ นี้ในทุก ๆ servlet url หากพบก็ให้ส่ง servlet นั้นไปที่ servlet ตัวอย่างที่ชื่อ invoker เพื่อทำงาน servlet นั้น

Servlet 2.2 เคยใช้ servlet aliases เป็นวิธีอ้างอิง servlets มาก่อน หากตรวจสอบดูไฟล์ web.xml ที่ไดเรกทอรี %CATALINA_HOME%/conf ของ Tomcat ตั้งแต่รุ่น 3.3 ถึง 4.1.7 จะพบว่ามีการใช้ servlet-mapping tag กำหนด url-pattern ที่ เป็น /servlet/* กับ servlet-name ชื่อ invoker ซึ่งก็หมายถึงถ้ามี servlet ใดมีชื่อนำหน้าเป็น servlet ให้ส่ง servlet นั้นไปทำงานที่ invoker

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.11.7 Context Path

ในการพัฒนาโปรแกรมใด ๆ เรามักไม่ต้องการเข้าไปทำงานในไคลเรททอรีของโปรแกรมเครื่องมือ หรือโปรแกรมของ Server เพื่อไม่ให้ส่งผลกระทบต่อไฟล์ของโปรแกรมเหล่านั้นโดยไม่ตั้งใจ ดังนั้น Tomcat จึงยอมให้เราทำ configuration เพื่อให้ไคลเรททอรีของ web apps อาจอยู่ที่ใดก็ได้

2.11.8 Servlet Lifecycle

คลาสของ servlet ไม่มี main() นั่นหมายถึงว่าทุก servlets ไม่มี thread สำหรับควบคุมการทำงานของตัวเอง เพราะ servlet container (ที่ทำงาน servlet นั้น) เป็นผู้ควบคุมวัฏจักรชีวิตของ servlet ซึ่งถูกกำหนดไว้แล้ว ผู้เขียนโปรแกรม servlets ไม่อาจกำหนดหรือเปลี่ยนแปลงวัฏจักรชีวิตของ servlet เอง ทำได้แต่เพียงกำหนดว่าเมื่อ servlet นั้นเข้าสู่สถานะใดจะให้ทำกิจกรรมใด

วัฏจักรชีวิตของ servlets แบ่งออกเป็นสถานะได้ดังนี้

- Unavailable ตอนที่ยังไม่มี servlets ของ instance ถูกทำลายไปแล้ว
- Instantiation ตอนที่ servlets ถูกโหลดเข้าสู่ servlet container จะสร้าง instance ของคลาส servlet ขึ้น (empty constructor จะถูกทำงาน)
- Initialization เมื่อสร้าง instance แล้ว servlet container จะเรียก init() เพื่อส่ง ServletConfig เป็นพารามิเตอร์ เพื่อนำไปกำหนดค่าเริ่มต้น
- Service เมื่อมี requests มีที่ servlet (ที่ถูก initialization แล้ว) servlet container จะเรียกมาที่ service() โดยส่ง request นั้นเข้ามาเป็นพารามิเตอร์ servletRequest และรับ response กลับเป็น ServletResponse
- Destroy เป็นตอนก่อนที่ instance ของ servlet นั้นถูกทำลาย servlet container จะเรียก destroy() ของ servlet นั้น

สรุปว่าผู้เขียนโปรแกรมสามารถกำหนดกิจกรรมให้ servlets ทำโดยสร้างเป็น method ที่เกี่ยวข้องกับการทำงานของ server เช่น invoker ก็จะถูก instantiated ตั้งแต่เมื่อ server เริ่มทำงาน แต่ servers บางยี่ห้อยอมให้ผู้ดูแลระบบสามารถเลือกระบุได้ว่าจะให้ servlets ใดถูก instantiated ตอน server เริ่มทำงาน หรือที่เวลานั้น ใช้ในกรณีที่เราต้องการให้ servlets นั้นถูก instantiated ไว้ก่อน เพื่อไม่ต้องเสียเวลาตอนจะใช้งาน

หาก servlets ใดไม่ได้ถูก instantiated ไว้ก่อน จะถูก instantiated เมื่อมี request มาใช้บริการเป็นครั้งแรก instantiated และ initialization แล้ว servlet container จะเก็บ instance นั้นไว้ในหน่วยความจำเพื่อให้บริการ requests ไปจนกว่าจะถูก destroy (ไม่ต้องสร้าง instance ขึ้นใหม่แต่ละ request) แสดงว่า servlets ที่ถูกโหลดแบบนี้จะใช้เวลาผกาน้อย ในการให้บริการครั้งแรก เพราะต้องทำ instantiated และ initialization แต่ในการเรียกใช้ครั้งต่อ ๆ ไปจะเร็วกว่า เพราะมี instance อยู่หน่วยความจำแล้ว

โดยปกติแล้ว servlets container จะทำการ destroy servlets ทั้งหมดที่สร้างไว้ เมื่อจะเลิกทำงาน แต่เนื่องจาก server ต้องให้บริการต่อเนื่องเป็นเวลานานจึงอาจมี servlets บางตัวที่มี instances อยู่ในหน่วยความจำโดยไม่ได้ถูกใช้งานเป็นเวลานาน ๆ ได้ เพื่อแก้ปัญหานี้ server บางยี่ห้อจึง อาจตัดสินใจทำการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

destroy servlets ที่ไม่ถูกใช้งานเป็นเวลานานได้เองโดยอัตโนมัติ หรือมีกลไกให้ผู้ดูแลระบบสามารถเลือก destroy servlets ที่ไม่ต้องการออกได้

ระหว่างที่ `init()` หรือ `destroy()` ถูกทำงาน servlet containers ต้องรับประกันว่าไม่มีการให้บริการ servlets นั้น (คือการทำงาน `service()`) ตลอดช่วงชีวิตของ servlets ใด ๆ `init()` และ `destroy()` จะถูกทำงานเพียงครั้งเดียวเท่านั้น ดังนั้นหากเราต้องการให้กิจกรรมใด (เช่น การกำหนดค่าเริ่มต้นต่าง ๆ) เกิดขึ้นก่อน servlet จะเริ่มให้บริการ ก็กำหนดไว้ใน `init()` และหากต้องการให้กิจกรรมใด (เช่น การกำหนดค่าเริ่มต้นต่าง ๆ) เกิดขึ้นก่อน servlet จะเริ่มให้บริการ ก็กำหนดไว้ใน `init()` และหากต้องการให้กิจกรรมใด (เช่น การปิดไฟล์ sockets หรือ connections รวมทั้งการเขียนใส่ไฟล์ log) เกิดขึ้นก่อน servlets จะหยุดให้บริการ ก็กำหนดไว้ใน `destroy()`

ใน `javax. Servlets package` มี `SingleThreadModel` เป็น tag interface (ไม่มี method) สำหรับให้คลาสของ servlets ทำการ implements เพื่อกำหนดให้มีเพียง thread เดียวเท่านั้นสามารถเข้าทำงาน `service()` ได้ที่เวลาหนึ่ง `SingleThreadModel` interface จะมีผลต่อพฤติกรรมของ servlets containers คือ

สำหรับ servlets ที่ไม่ได้ implements `SingleThreadModel` แล้ว servlets containers จะสร้างเพียงหนึ่ง instance สำหรับ servlet นั้น เมื่อมี request หนึ่งมาที่ servlet นั้น servlets containers จะใช้ thread เส้นหนึ่งเรียกทำงาน `service()` โดยไม่สนใจว่าที่เวลานั้นมี thread เส้นอื่น ๆ ทำงาน `service()` อยู่หรือไม่ แสดงว่าที่เวลาหนึ่งอาจมี thread มากกว่าหนึ่งเส้นทำงาน `service()` นั้น ดังนั้น servlets ประเภทนี้ต้องทำงานแบบ stateless คือการทำงานของ thread หนึ่งไม่ส่งผลกระทบต่อการทำงานของอีก thread หนึ่ง

ถ้าเป็น servlets ที่ implements `SingleThreadModel` แล้ว servlets containers อาจมีทางเลือกที่จะสร้าง queue ของ requests เพื่อให้แต่ละ request สร้าง thread ขึ้นใช้งาน `service()` ทีละคน หรืออาจสร้าง instances ของ servlets นั้นขึ้นมาหลาย ๆ ตัวและเก็บใส่ pool ไว้ เมื่อมี requests มาใช้งาน servlets นั้นพร้อม ๆ กัน ก็ให้แต่ละ request สร้าง thread ขึ้นใช้งาน `service()` แต่ละ instance ใน pool นั้น เพียงแต่ต้องรับประกันว่ามีเพียง thread เส้นเดียวที่ทำงาน `service()` ของ instance หนึ่ง

2.11.9 GenericServlet

คลาสของ servlets ต้องทำการ implements methods ทั้งหมดที่กำหนดโดย servlet interface แต่ method เหล่านี้มีเพียง `service()` เท่านั้นที่สำคัญต่อการให้บริการของ servlets ดังนั้นใน `javax.servlet.package` จึงมีคลาส `GenericServlet` ที่ทำการ implements servlet interface โดย implements `init()`, `destroy()`, `getServletInfo()` และ `getServletConfig()` ที่กำหนดใน servlets interface ไว้ให้เท่าที่จำเป็น ยกเว้น `service()` เป็น abstract method จึงทำให้ `GenericServlet` เป็น abstract class เราสามารถใช้คลาส `GenericServlet` สร้างคลาสสำหรับ servlet ได้โดยขยเป็นคลาสลูก แล้วอิมพลิเมนต์เฉพาะ `service()`

คลาส `GenericServlet` ทำการ implements `init(ServletConfig sc)` ของ Servlet interface ให้รับค่า `ServletConfig sc` มาเก็บไว้เป็น data member เหมือนกับที่แสดงในตัวอย่าง `MyServlet.java` ดังนั้นหากเราสร้างคลาสลูกของ `GenericServlet` และมีการ override `init(ServletConfig sc)` ก็ควรจะมีการเรียก `super.init(sc)` เพื่อให้ `ServletConfig sc` ถูกเก็บไว้เป็น data member

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.12 Threads

ระบบปฏิบัติการ UNIX, windows และ WindowsNT จะสามารถรับโปรแกรมเข้าไปทำงานได้มากกว่าหนึ่งโปรแกรมในเวลาเดียวกัน ทั้งที่อาจจะมีหน่วยประมวลผลเพียงตัวเดียว คุณสมบัติของระบบปฏิบัติการอย่างนี้เรียกว่า multitasking เป็นกลไกที่ช่วยให้เราใช้งานหน่วยประมวลผลได้อย่างมีประสิทธิภาพมากขึ้น ภาษาสำหรับโปรแกรมพยายามพัฒนาให้ผู้เขียนโปรแกรมสามารถใช้ประโยชน์จากโปรแกรมนี้ได้โดยตรง เรากำหนดว่า

โปรแกรม คือลำดับของคำสั่งชุดหนึ่งเป็นสิ่งที่ต้องถูกกระทำ (Passive)

Process คือกิจกรรมที่เป็นผลมาจากการทำงานของโปรแกรมเป็นสิ่งที่ทำงานได้ (active)

ระบบปฏิบัติการโดยทั่วไป มีกลไกสำหรับการจัดโปรแกรมให้หน่วยประมวลผลทำงาน โดยโหลดโปรแกรมที่ทำงานเข้าสู่หน่วยความจำ แล้วจัดการทำโปรแกรมเหล่านั้นได้ทำงานในหน่วยประมวลผล เมื่อโปรแกรมทำงานแล้วกิจกรรมของมันคือ Process ระหว่างที่ process ทำงานอยู่ระบบปฏิบัติงานต้องมีข้อมูลเกี่ยวกับ process นั้นเก็บในพื้นที่หน่วยความจำที่เรียกว่า process control block (PCB) โดยทั่วไปเป็น PCB จะประกอบด้วย โปรแกรมของ processor นั้น ตำแหน่งของคำสั่งต่อไปจะถูกทำงาน (program counter) สถานะของ process และพื้นที่หน่วยความจำที่เรียกว่า data segment สำหรับเก็บค่าตัวแปรค่าคงที่ หรือค่าที่ได้จากการคำนวณใน process นั้น

ในระบบปฏิบัติงานแบบ Multitasking ผู้ใช้ (อาจจะมากกว่าหนึ่งคน) สามารถส่งโปรแกรมให้แกระบบปฏิบัติงานมากกว่าหนึ่งโปรแกรมในเวลาเดียวกัน

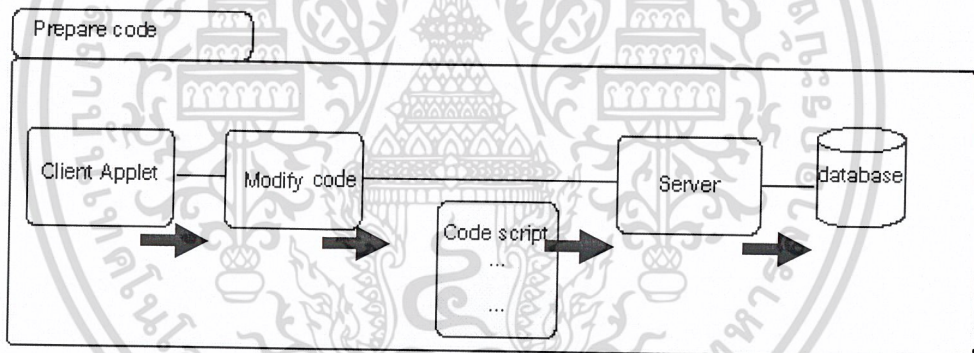
บทที่ 3

การออกแบบและการสร้างเกม

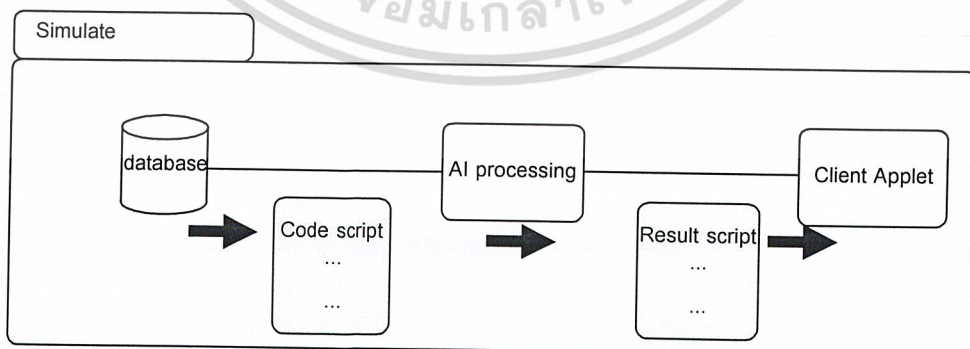
3.1 ภาพรวมของเกม

ในส่วนของเกมภาพรวมของโปรแกรมนี้จะขอแสดงให้เห็นถึง 2 ขั้นตอน คือการเตรียม Code (Prepare code) และการแสดงผลการรัน (Simulate) โดยแต่ละช่องโหว่ของเกมจะอธิบายได้ดังนี้

- Client Applet เป็นส่วนของหน้าจอของผู้เล่นในการจัดการ
- Modify Code ในส่วนของการกำหนดค่าทำทางการเคลื่อนไหวและทำทางการต่อสู้ให้กับ
- Code Script ส่วนของ script ที่ได้มาจากค่าที่เราได้กำหนดค่าจากการ modify
- Server เซิร์ฟเวอร์ที่ทำการประมวลผลการรันโค้ด
- Database ส่วนที่เก็บข้อมูลการ modify ของตัวละครที่ได้กำหนดไว้ถูกนำมาเก็บไว้ในฐานข้อมูล



รูปที่ 3.1 ภาพรวมของการสร้าง code script ที่เก็บเพื่อส่งไปยังเซิร์ฟเวอร์

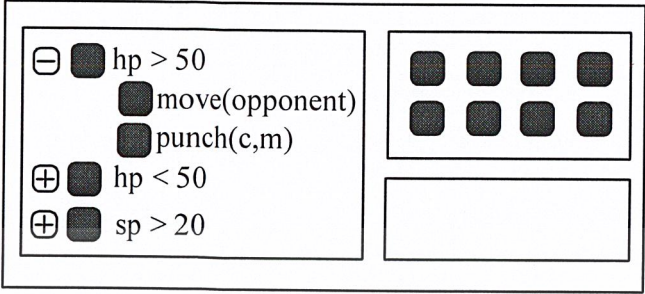


รูปที่ 3.2 ภาพรวมการนำ code script มาประมวลผลแล้วส่งผลลัพธ์กลับไปยัง client

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.1.1 การสร้าง Code script

แต่ละ client จะทำการ modify ai ของตนเองแล้วนำไปเก็บไว้ที่ server



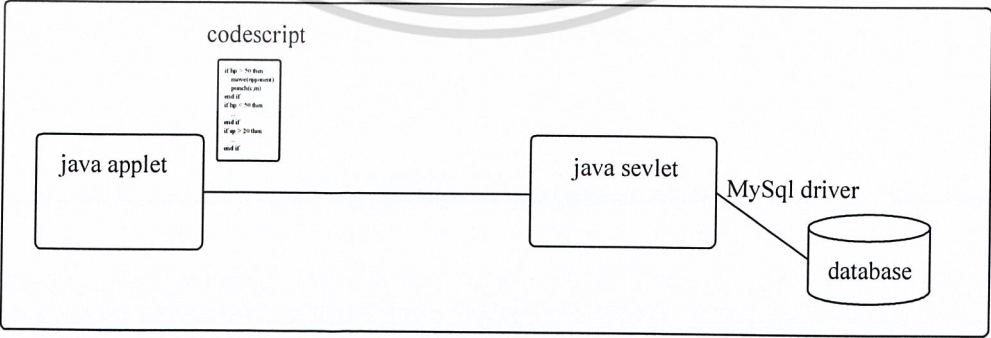
รูปที่ 3.3 ภาพแสดงการ modify AI โดยกำหนดลำดับการต่อสู้ของหุ่นยนต์

เมื่อทำการ modify เรียบร้อยแล้วจะได้ code script

```
if hp > 50 then
  move(opponent)
  punch(c,m)
end if
if hp < 50 then
  ...
end if
if sp > 20 then
  ...
end if
```

รูปที่ 3.4 แสดงตัวอย่าง code script ที่ได้จากการ modify AI

จากนั้น code script นี้จะถูกนำไปเก็บไว้ที่ server (ใน database ของ account ตนเอง)

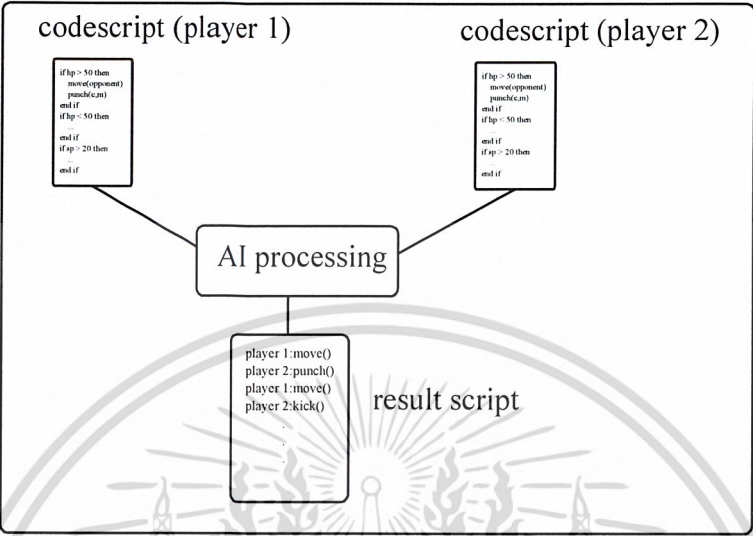


รูปที่ 3.5 แสดงการนำเอา code script ไปเก็บไว้ที่ server

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

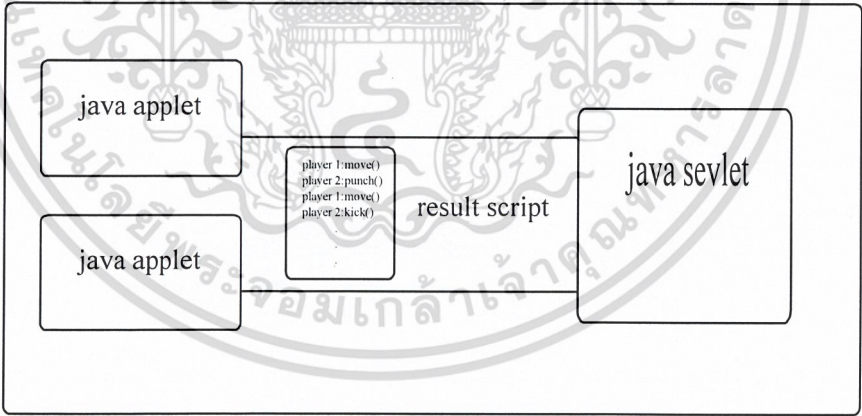
3.1.2 ประมวลผล Code script

เมื่อถึงเวลาต่อสู้ Server จะนำ code script ของแต่ละ account มาทำการประมวลผล



รูปที่ 3.6 แสดงการนำเอา code script แต่ละฝ่ายมาทำการประมวลผล
ซึ่งจะได้ผลลัพธ์เป็น result script

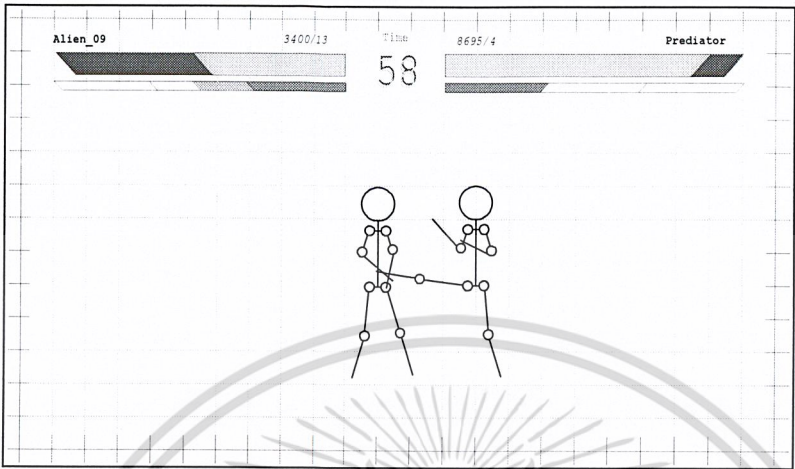
จากนั้น result script จะถูกส่งไปยัง client ทั้งสอง



รูปที่ 3.7 แสดงการส่ง result script กลับไปยัง client เพื่อนำไปแสดงผล

3.1.3 การแสดงผล (Result script)

เมื่อ client ได้รับ result script แล้วก็จะนำไปแสดงออกเป็นผลลัพธ์ของการต่อสู้



รูปที่ 3.8 แสดงการนำเอา result script มาแสดงผล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2 Function ที่สามารถกำหนดลงไป AI ได้

Control statements

- If condition

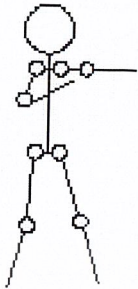
Primitive Action

- Move
- Punch
- Kick
- Guard
- Stand
- Special



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

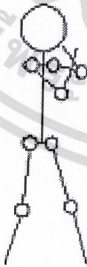
ภาพจำลองตัวละคร พร้อมแสดงถึงฟังก์ชันต่างๆที่สามารถทำงานได้

	function	damage
	Punch()	5
	Special()	15

รูปที่ 3.9 แสดงฟังก์ชัน *punch* และ *special*

	function	damage
	Kick()	10

รูปที่ 3.10 แสดงฟังก์ชัน *kick*

	function
	Guard()

รูปที่ 3.11 แสดง ฟังก์ชัน *guard*

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.3 การออกแบบฐานข้อมูล

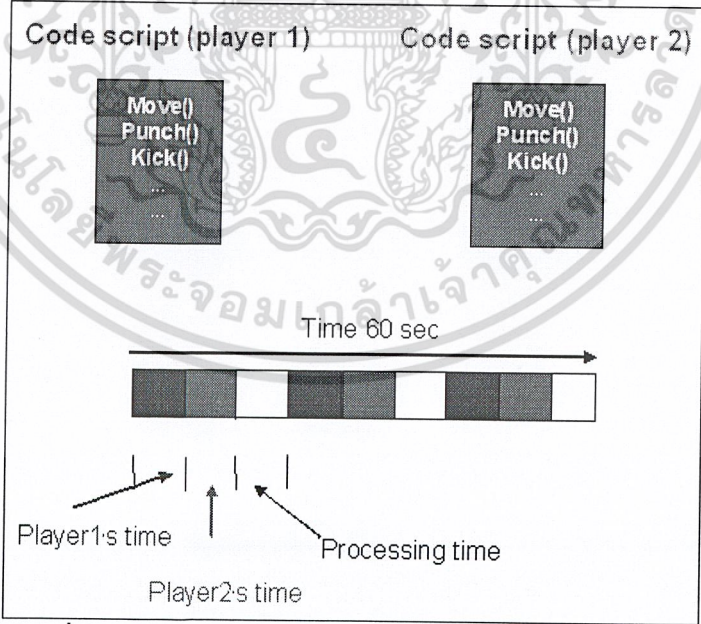
ในส่วนของผู้ server ได้ออกแบบ Table ที่ใช้ในโปรแกรม โดยมี Schema ดังรูป เพื่อใช้ในการเก็บข้อมูลที่จำเป็นต่างๆที่ใช้

Schema name	Data Type
Name	Varchar (12)
Password	Varchar (12)
Email	Varchar (20)
Script	Longblob

รูปที่ 3.12 แสดง Schema ที่ได้ออกแบบไว้ในโปรแกรม

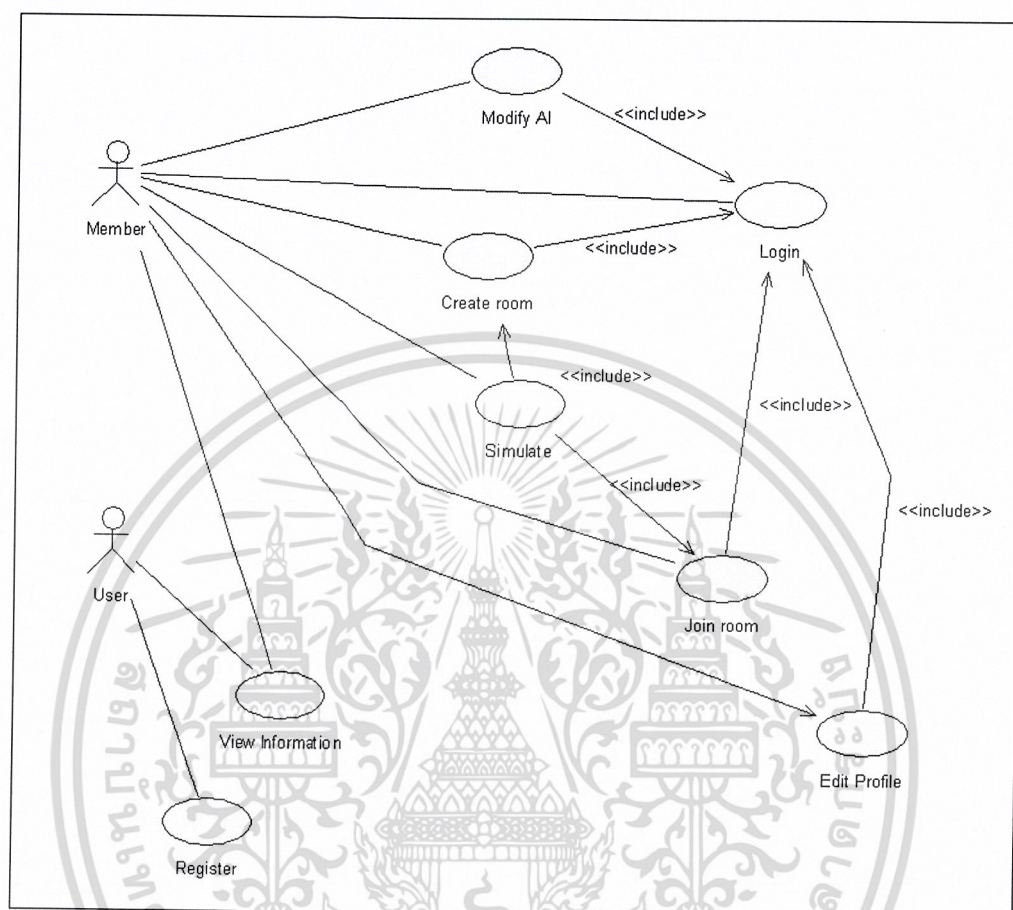
3.4 การประมวลผลทางฝั่ง server

การสร้าง result script เราทำการนำเอา script ทั้งสองฝั่งมาทำการประมวลผลจากนั้นนำมาสร้างเป็น result script โดยอธิบายโดยรูปที่ 3.13 โดยการประมวลผลจะใช้เวลา 60 วินาที โปรแกรมจะทำการดึงเอา code script ของแต่ละฝั่งขึ้นมาจากนั้นนำมาประมวลผล โดยการประมวลผล code processing จะทำการดึงเองคำสั่ง code script ของแต่ละฝั่งขึ้นมาทีละ 1 คำสั่งแล้วนำมาเปรียบเทียบกับว่าหลังจากทั้งสองฝ่ายทำคำสั่งแล้วผลลัพธ์ออกมาเป็นอย่างไรก็จะนำมาสร้างเป็น result script



รูปที่ 3.13 แสดง การนำเอา code แต่ละฝั่งมาประมวลผล

3.5 แผนภาพ Use case diagram



รูปที่ 3.14 แผนภาพ Use case diagram แสดงภาพรวมของเกม

use case ของเกมนั้นได้อธิบายถึงภาพรวมของการพัฒนา ซึ่งจะเห็นได้ว่าเกมนั้นมี actor และ use case โดยจะอธิบายแยกได้ดังนี้

1. actor มี 2 คนด้วยกัน

- actor ที่เป็น user หรือผู้ใช้ทั่วไป ที่มีหน้าที่ในการใช้งานเกมจำกัดเฉพาะบางส่วนเท่านั้น ที่เข้าใช้งานได้เฉพาะส่วนของการสมัครสมาชิก (Register) และการดูข้อมูลรายละเอียดในตัวเกม (View Information)
- actor ที่เป็น Member หรือสมาชิก ที่สามารถเข้าใช้งานได้ในเกือบทุก use case ของเกม โดยที่ก่อนจะทำการเป็นสมาชิกนั้นต้องผ่านการสมัครสมาชิกก่อน

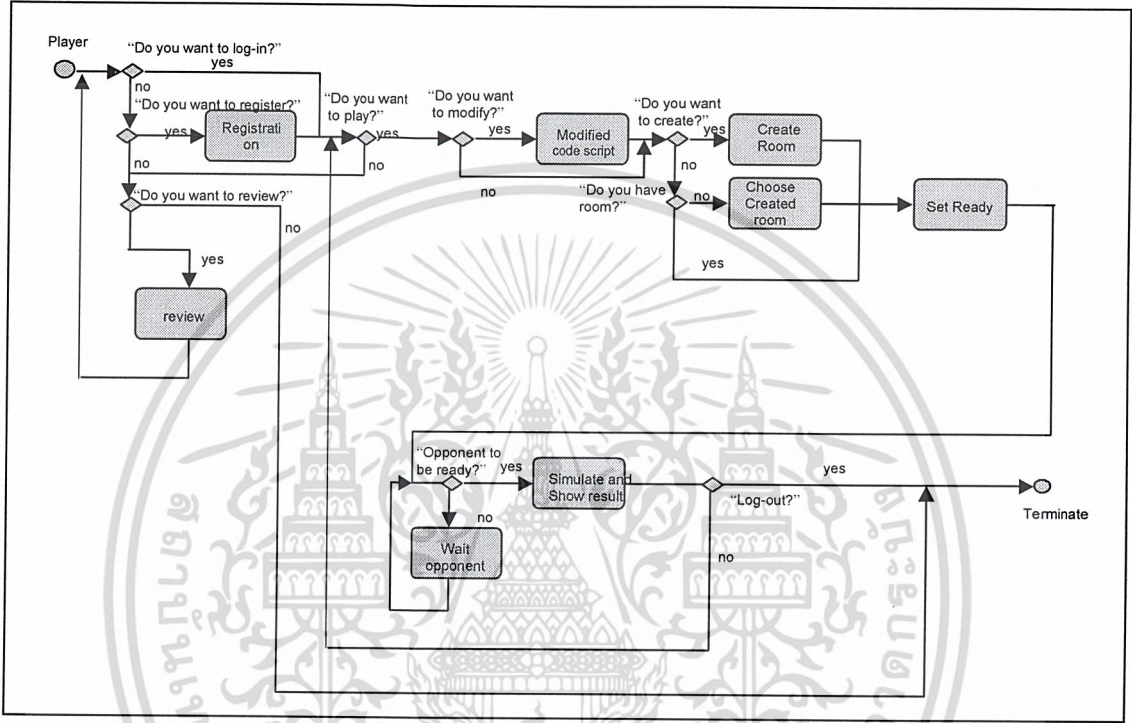
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. use case มีรายละเอียดดังต่อไปนี้

- **Register** เป็นการสมัครสมาชิกในการเข้าเล่นเกมนั้นต้องผ่านการสมัครสมาชิกเพื่อสร้าง user name และ password ก่อน ถึงจะสามารถเข้าใช้งานและเล่นเกมได้
- **View Information** ดูข้อมูลการรายละเอียดของเกม โดยส่วนนี้จะเป็นส่วนของเว็บไซต์ที่จะมีข้อมูลต่างๆเพื่อการเล่นเกม
- **Modify Code** เป็นการที่ผู้เล่นกำหนดค่าการต่อสู้ให้กับตัวละครของตัวเอง ซึ่งจะกำหนดท่าทางการต่อสู้ และการเคลื่อนไหวได้
- **Login** ก่อนการจะทำการใดๆเกี่ยวกับเกมในฐานะสมาชิกนั้น ต้องทำการ log in หรือเข้าสู่ระบบเสียก่อน ซึ่งในการ log in นั้นต้องอาศัย user และ password ที่ได้ทำการ Register ไว้ก่อนแล้ว
- **Create room** ทำการสร้างห้องขึ้นมาเพื่อเล่นเกม ในกรณีที่ไม่ต้องการเข้าไปเล่นกับห้องที่คนอื่นสร้างไว้ จึงทำการสร้างห้องขึ้นมาใหม่เพื่อรอผู้ต่อสู้อีกฝ่ายเข้ามาทำการเล่นเกมด้วย
- **Simulate** ทำการประมวลผลการเล่นเกม ซึ่งผู้เล่นทั้งสองฝ่ายได้ทำการตกลงยืนยันค่า Code ซึ่งได้กำหนดไปแล้วจะทำการประมวลผลมาแสดงที่หน้าจอของผู้เล่นแต่ละคน
- **Join room** คือมีผู้สร้างห้องเพื่อรออีกฝ่ายเข้ามาเล่นแล้ว สามารถเข้าไป Join หรือร่วมเล่นเกมได้
- **Edit Profile** ทำการแก้ไขข้อมูลของผู้เล่น ซึ่งเป็นข้อมูลที่ได้ทำการสมัครสมาชิกไปในตอนแรก

3.6 แผนภาพ Activities Diagram

ไคอะแกรมนี้แสดงถึงภาพรวมของทั้งระบบที่แสดงให้เห็นลำดับขั้นตอนการเข้าใช้งานเกม
ไคอะแกรมนี้นี้ช่วยให้ผู้ที่ต้องการศึกษาโครงสร้างหรือภาพรวม สามารถเข้าใจได้ง่ายขึ้นเพราะง่ายต่อการทำความเข้าใจ

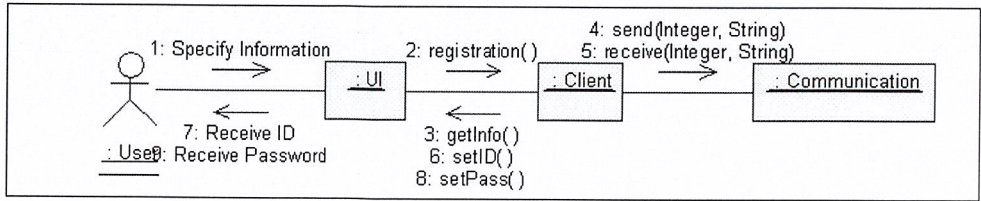


รูปที่ 3.15 Activity Diagram ของเกม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.7 แผนภาพ Collaboration diagram

3.7.1 Register

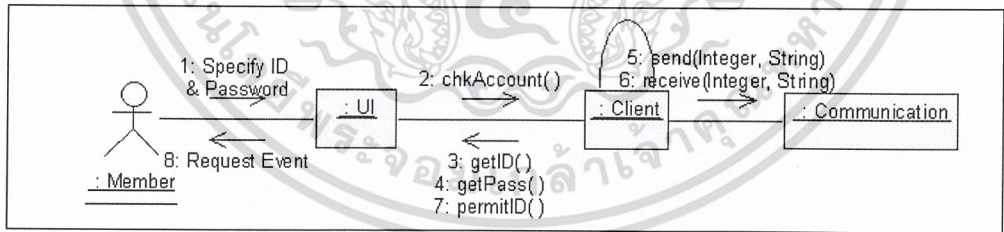


รูปที่ 3.16 Collaboration Diagram ของการ Register

ขั้นตอนในการเข้าสมัครสมาชิก(Register)

- 1. Specify Information ทำการระบุข้อมูลของผู้ลงทะเบียน
- 2. registration() ทำการยืนยันการลงทะเบียน
- 3. getInfo() รับข้อมูลการลงทะเบียน
- 4. send(Integer, String) ส่งข้อมูลไปยังเซิร์ฟเวอร์
- 5. receive(Integer, String) รับค่าข้อมูลกลับมาจากเซิร์ฟเวอร์
- 6. setID() กำหนดค่าชื่อผู้ใช้
- 7. ReceiveID user รับค่าชื่อผู้ใช้
- 8. setPass() กำหนดค่ารหัสผ่าน
- 9. Receive Password user รับค่ารหัสผ่าน

3.7.2 Log in



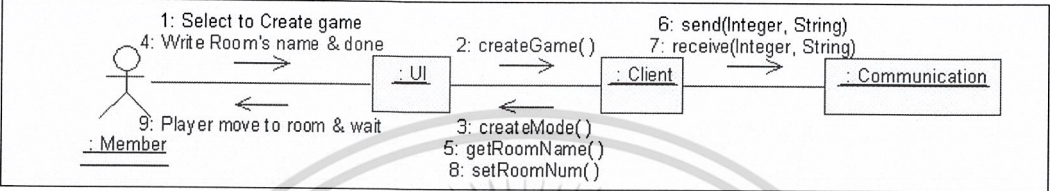
รูปที่ 3.17 Collaboration Diagram ของการ Login

ขั้นตอนในการสร้างเกม(Log in)

- 1. Specify ID&Password ขั้นตอนที่ทำกรระบุชื่อผู้ใช้และรหัสผ่าน
- 2. chkAccount() ทำการตรวจสอบว่ามีชื่อผู้ใช้ในระบบหรือไม่
- 3. getID() Client รับชื่อผู้ใช้จากสมาชิก
- 4. getPass() Client รับชื่อรหัสผ่านจากสมาชิก

- 5. send(Integer,String) ส่งค่าการร้องไปยัง server
- 6. receive(Integer,String) รับค่าการร้องขอจาก server
- 7. permitID() ยืนยันข้อมูล
- 8. Request Event รอรับการเลือกการเล่นจาก user

3.7.3 Create Room

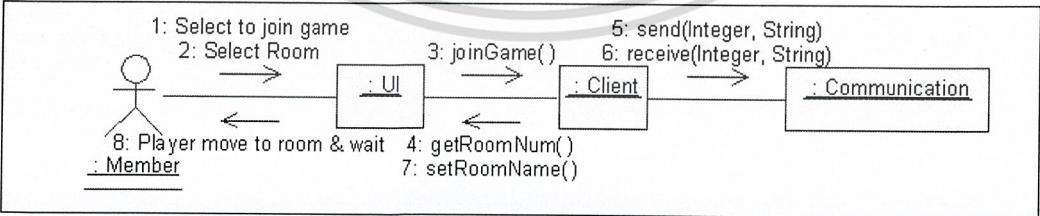


รูปที่ 3.18 Collaboration Diagram ของการ Create Game

ขั้นตอนในการสร้างห้อง (Create room)

- 1. Select to Create room ทำการเลือกเมนูเพื่อสร้างห้อง
- 2. createGame() ทำการเลือกสร้างห้อง
- 3. createMode() สร้างโหมด
- 4. Write Room's name & done กำหนดชื่อห้องที่ต้องการสร้าง
- 5. getRoomName() รับชื่อห้องที่ได้กำหนดให้
- 6. send(Integer, String) ส่งค่าการร้องไปยัง server
- 7. receive(Integer, String) รับค่าการร้องขอจาก server
- 8. setRoomNum() ระบบคืนค่าเลขประจำห้องที่สร้างขึ้นมา
- 9. Player move to room & wait ผู้เล่นเข้ายังห้องที่ได้สร้างและรออีกฝ่ายเขามาร่วมเล่นเกม

3.7.4 Join Game



รูปที่ 3.19 Collaboration Diagram ของการ Join Game

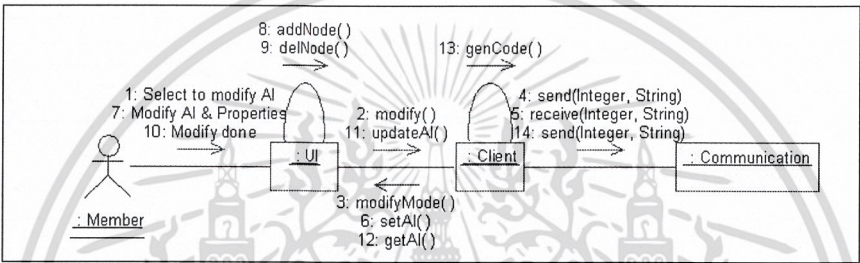
ขั้นตอนในการเข้าเล่นเกม (Join Game)

- 1. Select to join game เลือกที่จะทำการ join game

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- 2. Select Room ทำการเลือกห้องที่ได้สร้างไปแล้วจากรายชื่อห้องที่มีอยู่
- 3. joinGame ทำการเลือก join game
- 4. getRoomNum() รับชื่อห้องที่ต้องการเข้าไป join
- 5. send(Integer, Integer) ส่งค่าไปยังเซิร์ฟเวอร์
- 6. receive(Integer, String) รับค่ามาจากเซิร์ฟเวอร์
- 7. setRoomName() กำหนดค่าหมายเลขห้อง
- 8. Player move to room & wait ผู้เล่นเข้าไปยังห้องเพื่อเล่นและคอยคู่ต่อสู้พร้อม

3.7.5 Modify Code

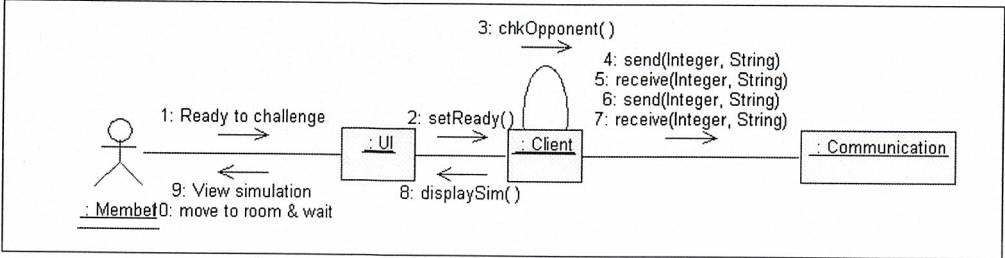


รูปที่ 3.20 Collaboration Diagram ของการ Modify code

ขั้นตอนในการกำหนดค่าการต่อสู้ (Modify code)

- 1. Select to modify AI ทำการเลือกเพื่อที่จะกำหนดค่าการต่อสู้ให้กับตัวละคร
- 2. modify() เลือกทำการ modify code
- 3. modifyMode() เปลี่ยนไปยังหน้าจอ modify code
- 4. send(Integer, String) ทำการส่งค่าไปยังเซิร์ฟเวอร์
- 5. receive(Integer, String) ทำการรับค่ามาจากเซิร์ฟเวอร์
- 6. setAI() นำค่า ai เดิมที่รับมาจาก server set ลงหน้าจอ
- 7. Modify AI & Properties
- 8. addNode()
- 9. delNode()
- 10. Modify done ทำการยืนยันการ modify ค่าให้กับตัวละคร
- 11. updateAI() ทำการ update ค่า code ลงบนฐานข้อมูล
- 12. getAI() ทำการรับค่า code ที่ได้กำหนดไป
- 13. genCode() แล้วทำการสร้าง code นั้นออกมาเพื่อส่งไปเก็บยังเซิร์ฟเวอร์
- 14. send(Integer, String) ส่งค่าต่างๆที่ได้กำหนดไปยังเซิร์ฟเวอร์

3.7.6 Simulation



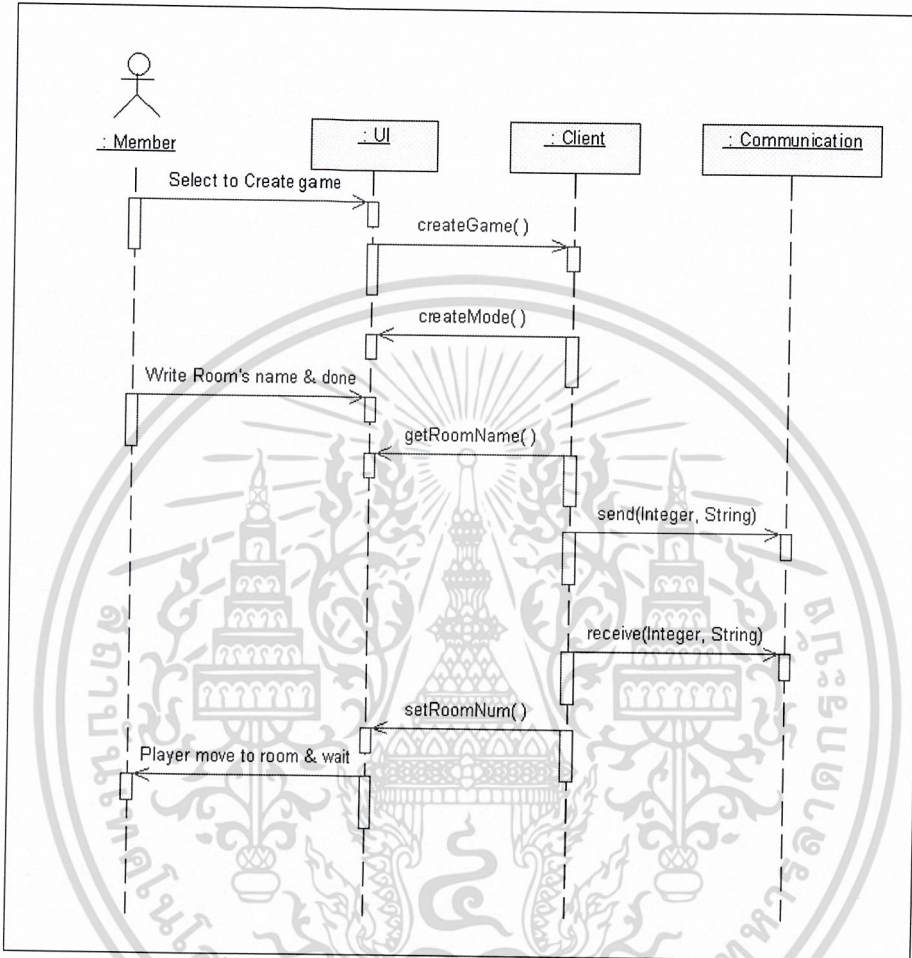
รูปที่ 3.21 Collaboration Diagram ของการ Simulation

ขั้นตอนในการแสดงผลการรัน Code (Simulation)

1. Ready to challenge กำหนดว่าพร้อมต่อสู้
2. setReady() กำหนดสถานะพร้อมจะเล่น
3. chkOpponent() ตรวจสอบความพร้อมของคู่ต่อสู้
4. send(Integer, String) ส่งค่าไปยังเซิร์ฟเวอร์ เพื่อให้ server ประมวลผล result script
5. receive(Integer, String) รับค่ายืนยันกลับมาจากเซิร์ฟเวอร์
6. send(Integer, String) ส่งค่าไปยังเซิร์ฟเวอร์เพื่อร้องขอ result script
7. receive(Integer, String) รับค่า result script มาจากเซิร์ฟเวอร์
8. displaySim() เรียกฟังก์ชันการแสดงผล
9. View simulation แสดงผลการรันยังเครื่องของผู้เล่น
10. move to room & wait กลับมายังห้องและสถานะคอย

3.8 แผนภาพ Sequence diagram

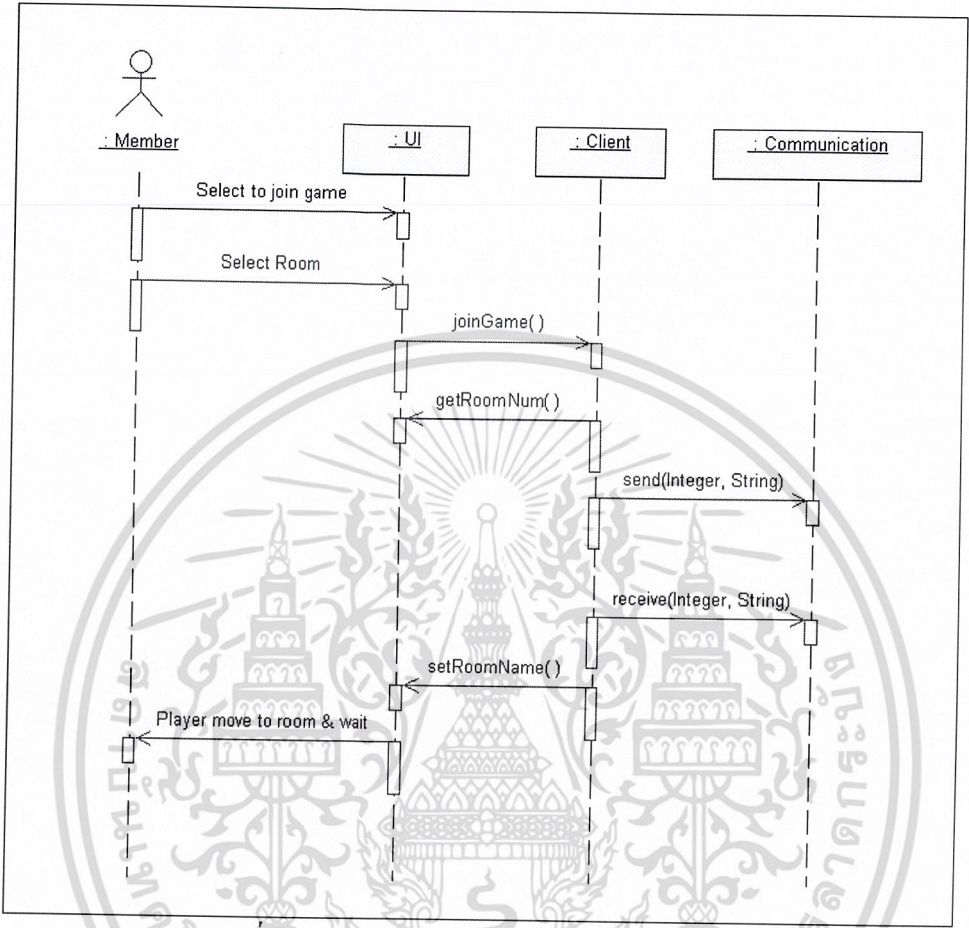
3.8.1 Create Game เป็นลำดับในการสร้างเกม



รูปที่ 3.22 Sequence Diagram ของการ Create Game

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

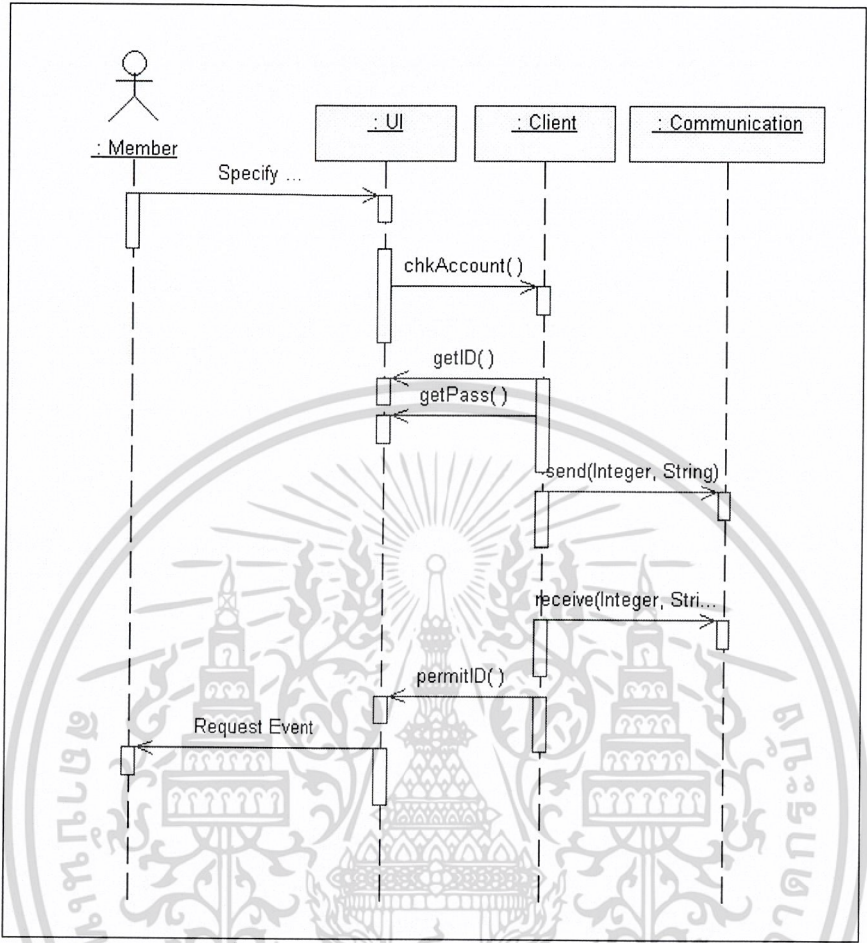
3.8.2 Join Game เป็นแผนภาพแสดงลำดับในการร่วมเข้าเล่นเกม



รูปที่ 3.23 Sequence Diagram ของการ Join Game

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

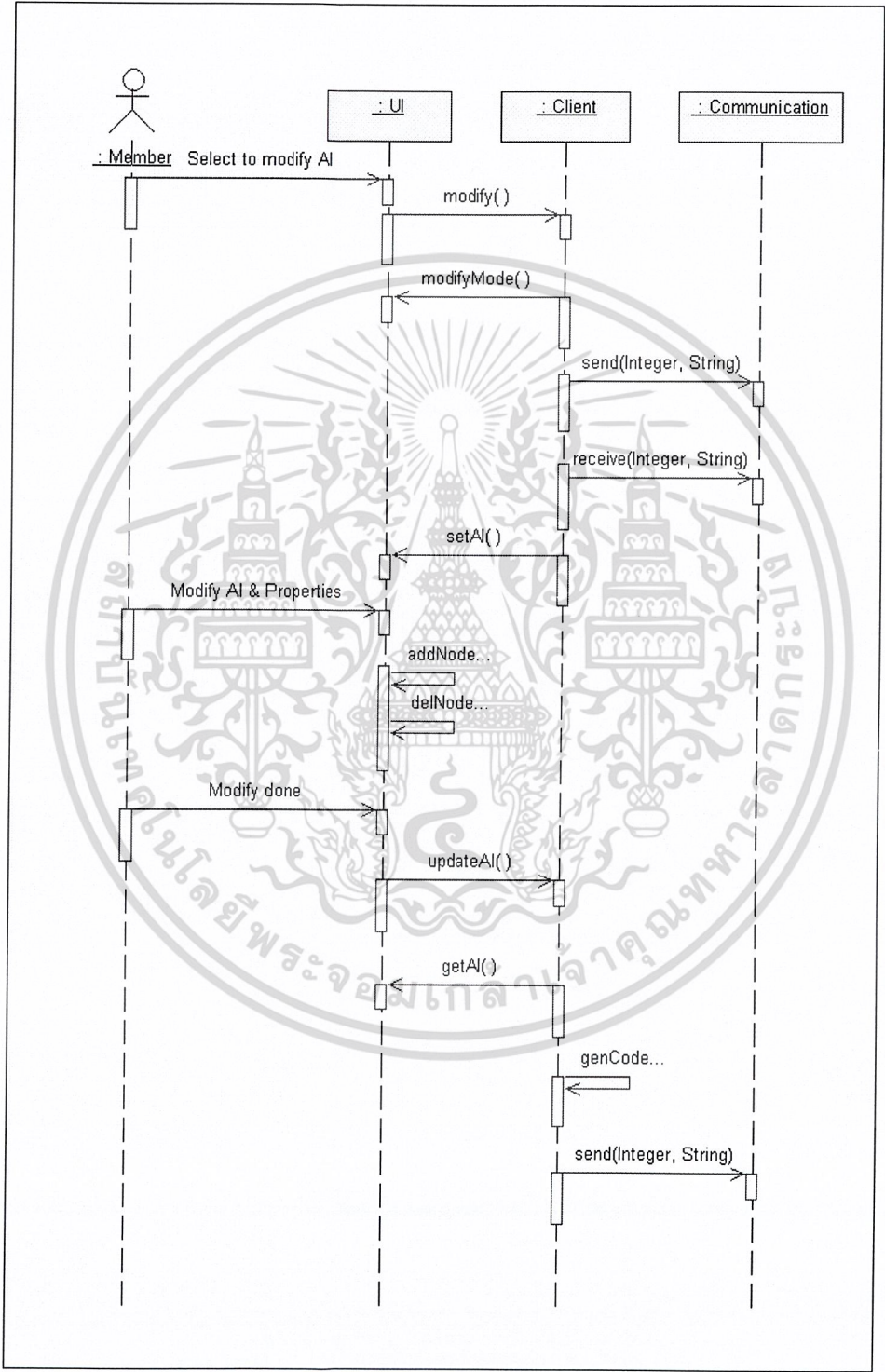
3.8.3 Log in เป็นแผนภาพแสดงลำดับในการล็อกอินเข้าสู่ระบบ



รูปที่ 3.24 Sequence Diagram ของการ Login

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

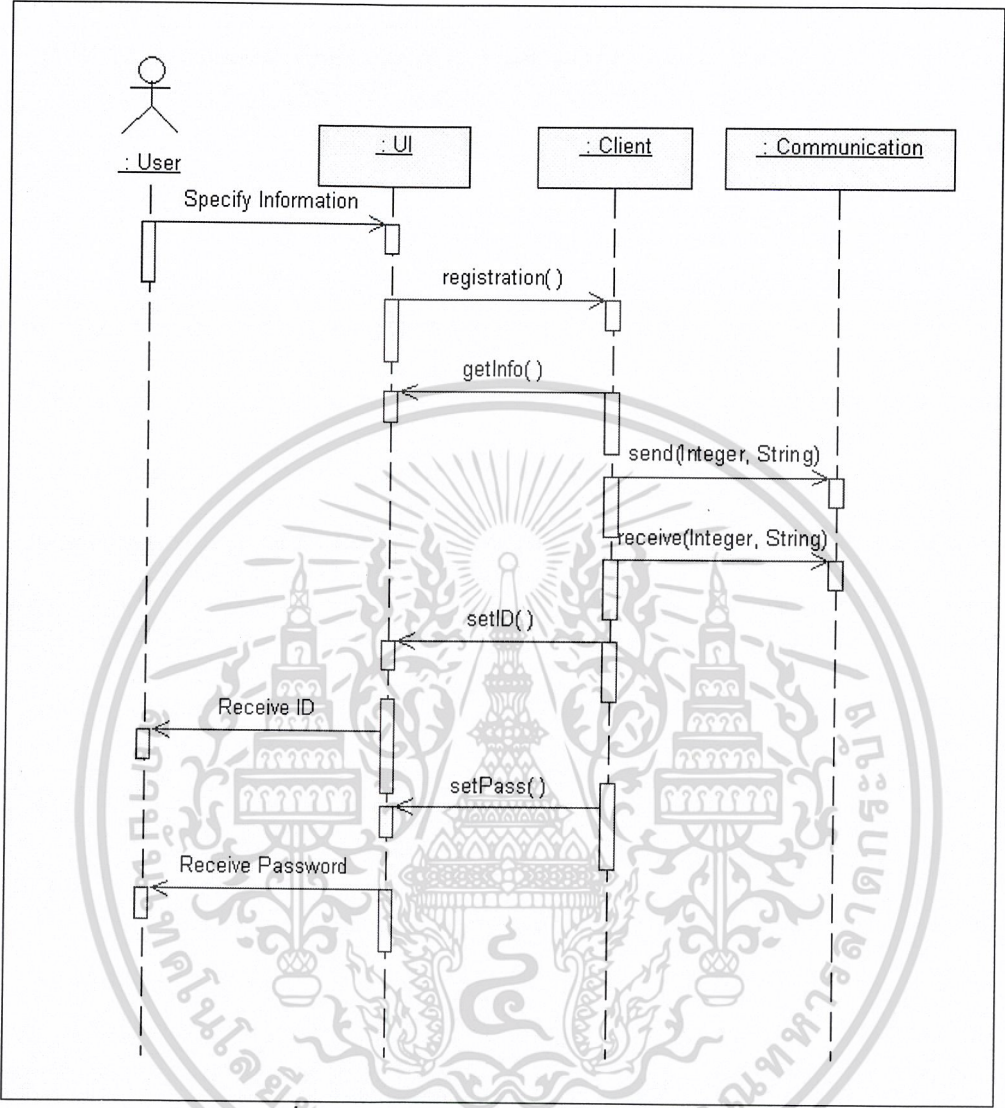
3.8.4 Modify code เป็นแผนภาพแสดงลำดับในการกำหนดค่าทางการต่อสู้และการเคลื่อนไหวให้ตัวละคร



รูปที่ 3.25 Sequence Diagramของการ Modify code

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

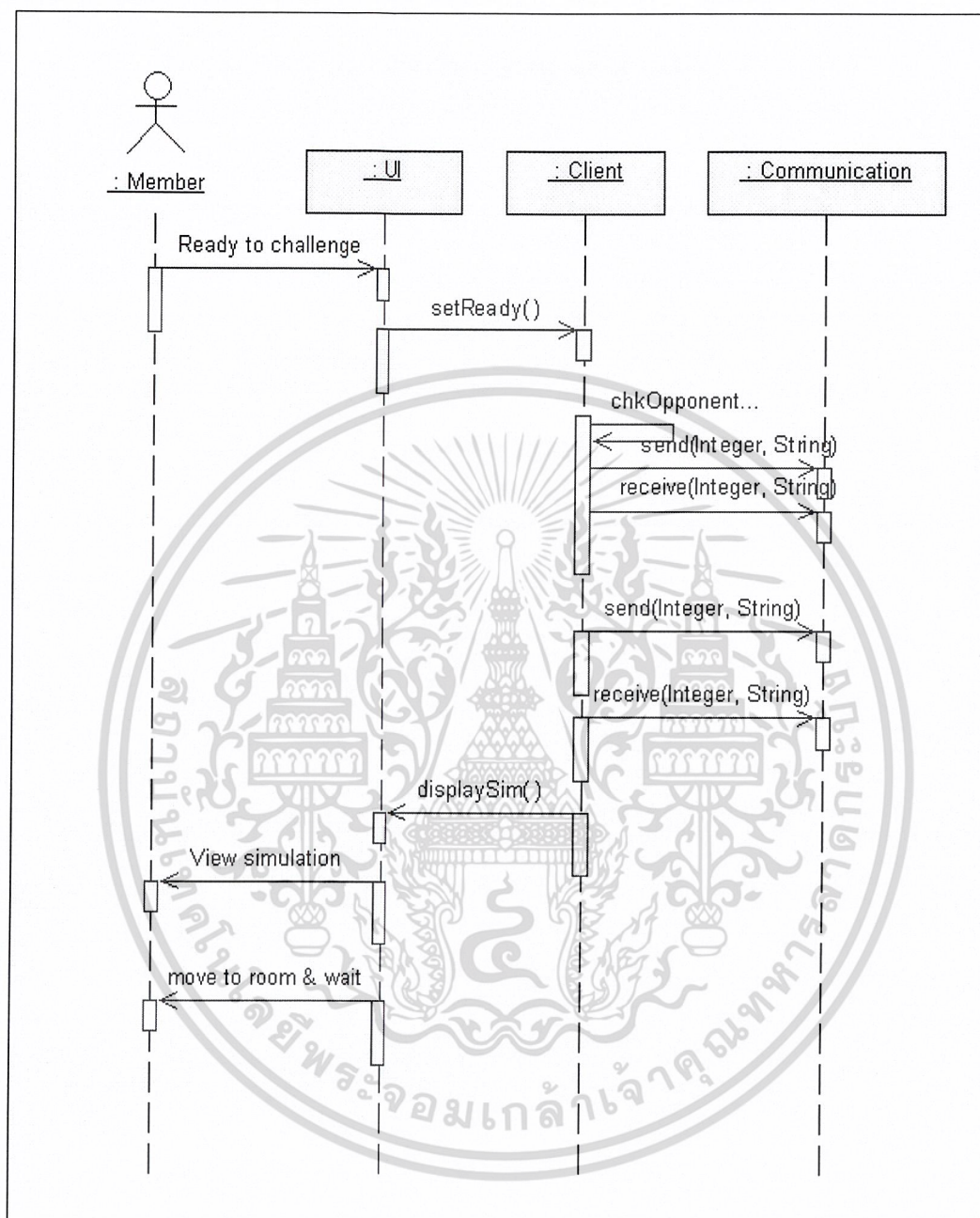
3.8.5 Register เป็นแผนภาพแสดงลำดับในการสมัครสมาชิก



รูปที่ 3.26 Sequence Diagramของกร Register

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.8.6 Simulation เป็นแผนภาพแสดงลำดับในการจะแสดงผลการรันเกมที่ได้รับมาจากทางเซิร์ฟเวอร์



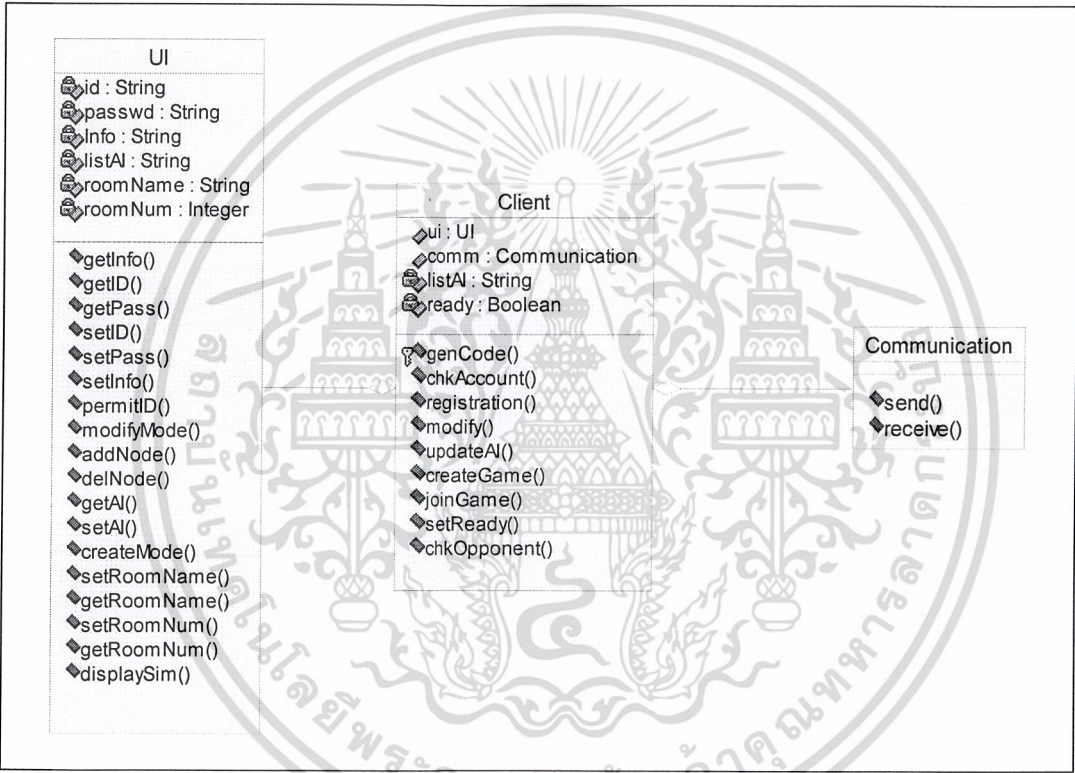
รูปที่ 3.27 Sequence Diagram ของการ Simulation

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.9 Class Diagram

Class Diagram จะมีทั้งหมด 3 โมดูลหลักคือ UI, Client และ Communication โดยที่สามารถอธิบายได้ดังนี้

- UI เป็นส่วนของหน้าจอการแสดงผลเพื่อ
- Client เครื่องผู้เล่นในการรับและจัดการค่าต่างๆ
- Communication ค่าที่ถูก



รูปที่ 3.28 Class Diagram

3.10 กฎในการเล่นเกม (Rules of game)

1. พลังชีวิต(HP) เริ่มต้นก่อนการเล่นกำหนดให้มีฝ่ายละ 100 หน่วย
2. ค่าความเสียหายที่เกิดขึ้นจากการต่อสู้มีดังนี้
 - a. ท่าต่อย(punch) ถ้าโดนฝ่ายตรงข้าม จะทำให้ฝ่ายตรงข้ามเสียพลังชีวิต 5 HP
 - b. ท่าเตะ (kick) ถ้าโดนฝ่ายตรงข้าม จะทำให้ฝ่ายตรงข้ามเสียพลังชีวิต 10 HP (แต่ถ้าใช้ถ้านี้แล้วโดนคู่ต่อสู้สวนกลับด้วย ท่าต่อย ท่าเตะ และ ท่าพิเศษ ผู้ใช้ท่านี้จะเสียพลังชีวิตเองด้วย 10 HP)
 - c. ท่าพิเศษ (special) ถ้าโดนฝ่ายตรงข้าม จะทำให้ฝ่ายตรงข้ามเสียพลังชีวิต 15 HP (ท่านี้สามารถใช้ได้เพียงครั้งเดียวต่อการต่อสู้หนึ่งรอบเท่านั้น)
 - d. เมื่อป้องกันโดยใช้ท่า Guard ค่า damage จะถูกหารด้วย 2 จากค่าความเสียหายนั้นๆ (ตามข้อ a-c) และ ถ้าโดนฝ่ายตรงข้ามใช้ ท่าต่อย,ท่าเตะ และ ท่าพิเศษ จะเสีย พลังชีวิตเพียง 2 HP
3. ผู้เล่นฝ่ายหนึ่งจะเอาชนะฝ่ายตรงข้ามภายในเวลาที่กำหนด โดยกำหนดให้เวลาในการแข่งขันเป็น 60 วินาที ซึ่งผลการแข่งขันมีดังนี้
 - a. เมื่อหมดเวลาการแข่งขัน
 - พลังชีวิตเป็นตัวกำหนดผลการแข่งขันโดยผู้เล่นฝ่ายไหนที่มีพลังมากกว่าจะเป็นผู้ชนะ
 - หากเวลาหมดแล้วพลังชีวิตเหลือเท่ากันถือว่าเสมอกัน
 - b. รู้ผลก่อนหมดเวลาการแข่งขัน
4. การสะสมคะแนนในส่วนของการ tournament นั้นผู้ใช้นั้นจะได้คะแนนสะสมครั้งละ 5 คะแนน, แต่จะไม่ได้คะแนน และเสมอ 1 คะแนน

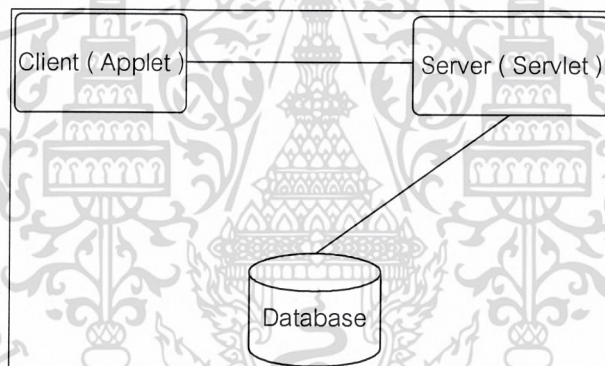
บทที่ 4

ผลการทดลองและทดสอบ

4.1 เครื่องมือที่ใช้ในการพัฒนา

ใช้ Eclipse เป็น IDE ซึ่งเป็น Open source และสามารถทำงานได้บนหลาย Platform สำหรับพัฒนาโปรแกรมภาษา Java ทั้งในส่วนที่เป็น Client คือเกมบน Java Applet 2D และในส่วนของ Server คือ Servlet/JSP โดยใช้ plug-in ของโปรแกรม Eclipse ชื่อ Lombok สำหรับช่วยในการเขียนโปรแกรมที่เป็น Servlet และ JSP

ได้ทำการทดลองการสื่อสารกันระหว่าง Client ซึ่งเป็น applet กับ server ซึ่งใช้ servlet โดยการส่ง request ต่างๆ ไปยัง servlet และ ทำการส่งข้อมูลไปเก็บไปยัง database ซึ่งใช้ MySQL

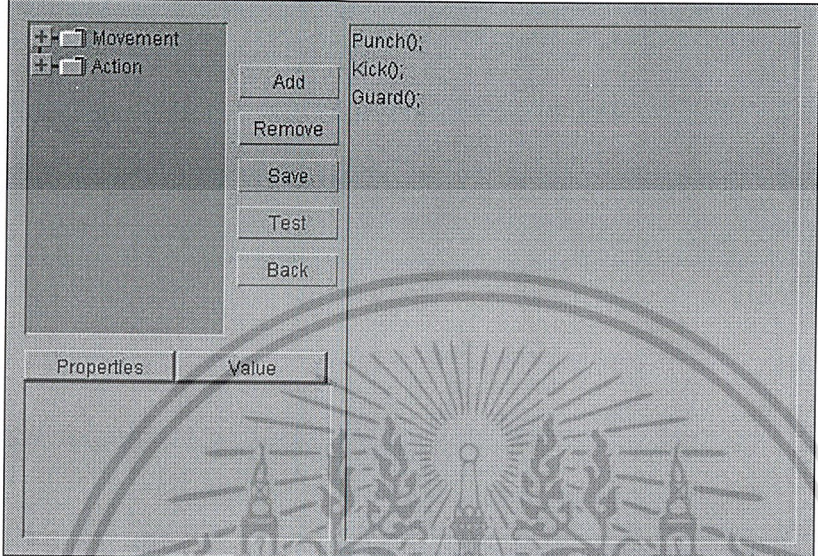


รูปที่ 4.1 แสดงการภาพรวมการทดลอง

4.2 ทำการทดลอง

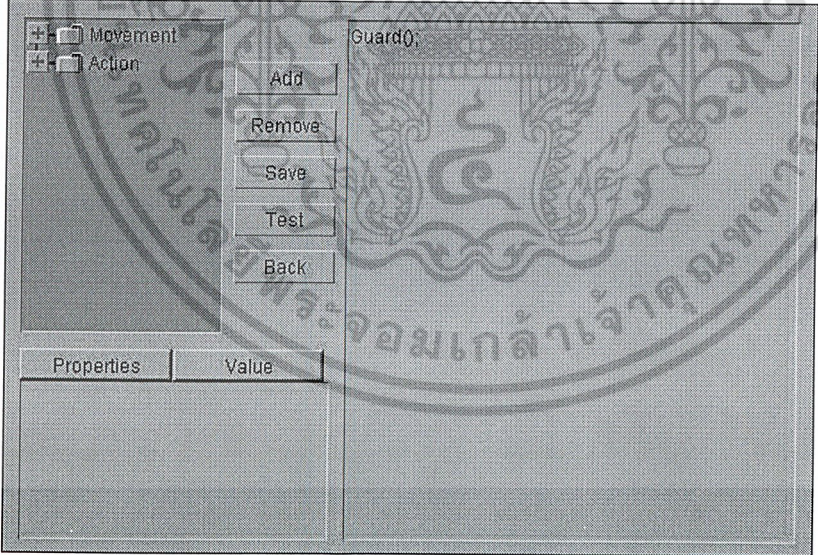
4.2.1 กรณีที่เล่นแล้วผลลัพธ์ออกมาเสมอ

1. จากหน้าจอทำการกำหนดค่าให้กับผู้เล่นชื่อ asd



รูปที่ 4.2 กำหนดค่าโดยผู้เล่นชื่อ asd

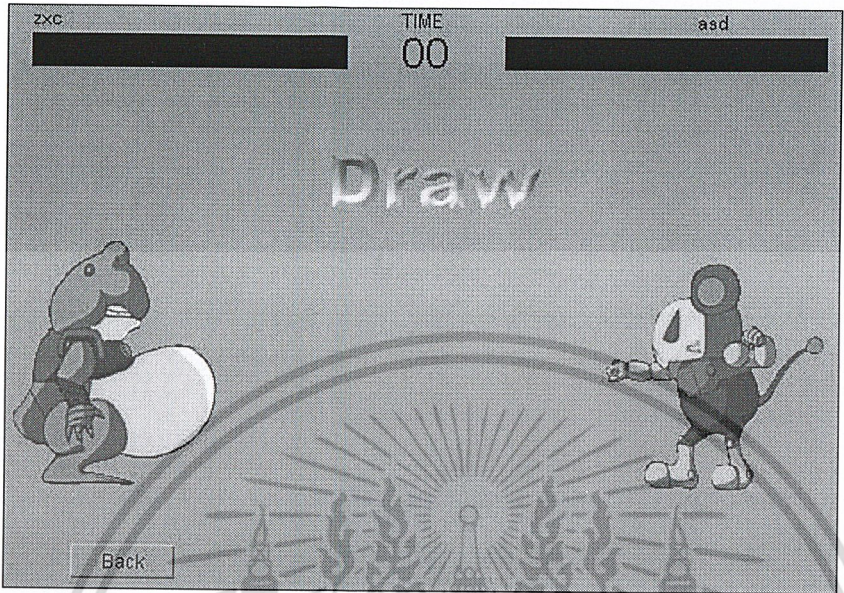
2. จากหน้าจอทำการกำหนดค่าให้กับผู้เล่นชื่อ zxc



รูปที่ 4.3 กำหนดค่าโดยผู้เล่นชื่อ zxc

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3. หน้าจอแสดงเมื่อกำหนดค่าและทำการรันเรียบร้อยแล้ว ผลปรากฏเสมอ (Draw)

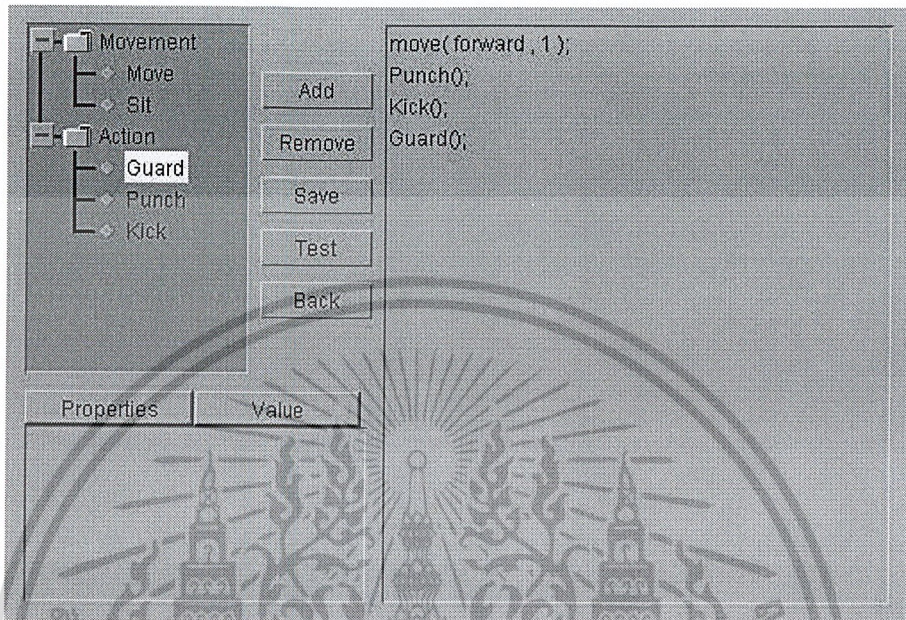


รูปที่ 4.4 แสดงผลการแข่งขันกรณีเสมอ (Draw)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

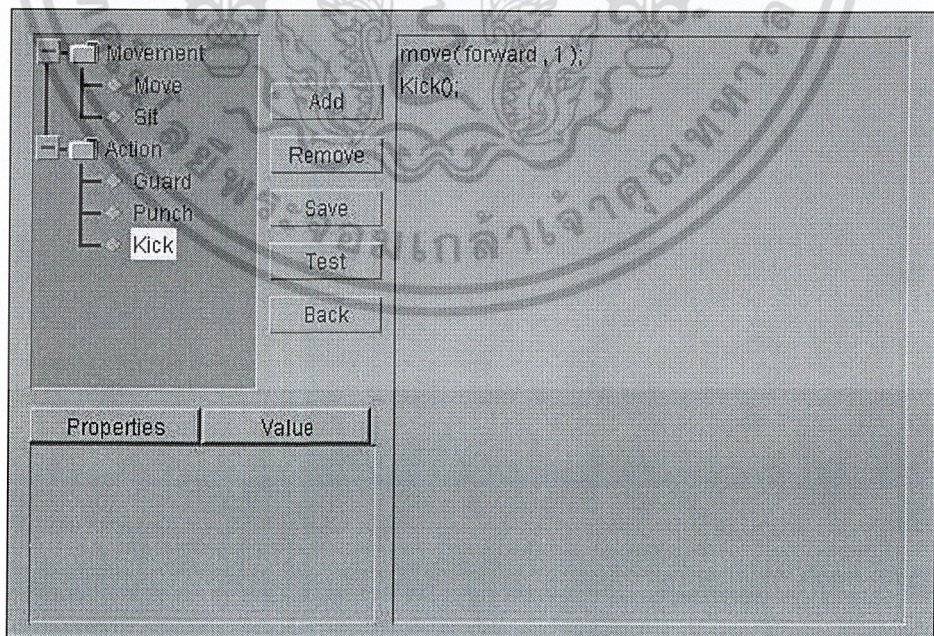
4.2.2 กรณีที่เล่นแล้วผลลัพธ์ออกมา แพ้ชนะ

1. ทำการกำหนด code โดยผู้เล่นชื่อ asd



รูปที่ 4.5 กำหนดค่าโดยผู้เล่นชื่อ asd

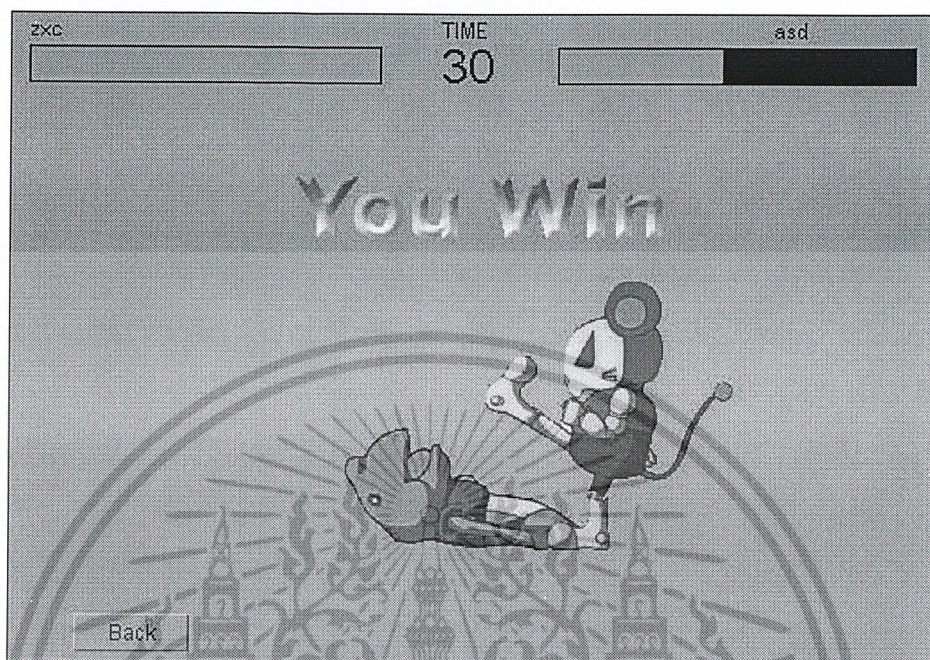
2. ทำการกำหนด code โดยผู้เล่นชื่อ zxc



รูปที่ 4.6 กำหนดค่าโดยผู้เล่นชื่อ zxc

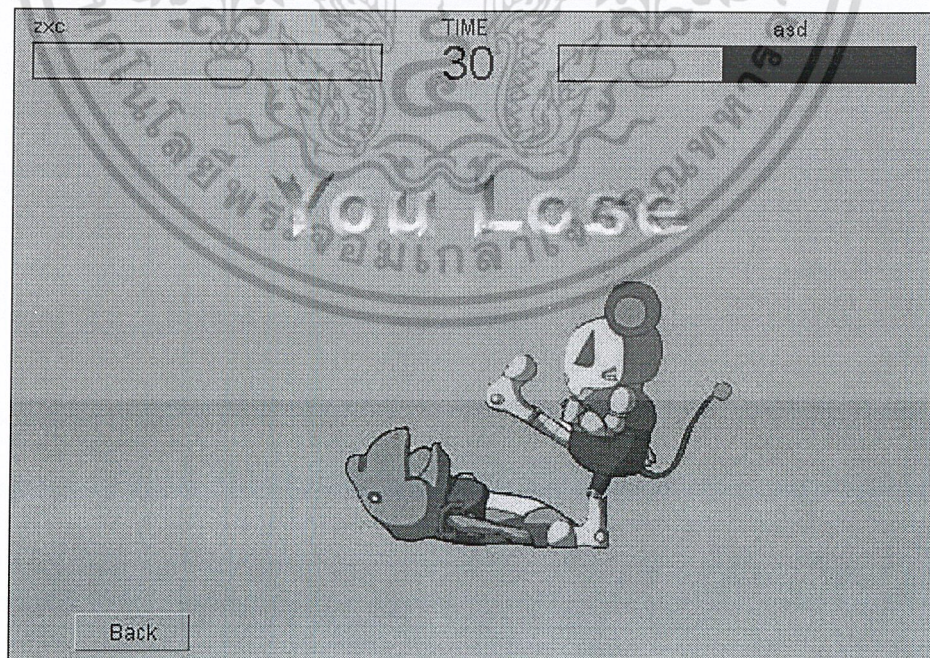
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3. ผู้เล่น asd เป็นฝ่ายชนะแสดงผลการรันข้อความว่า “You Win”



รูปที่ 4.7 แสดงผล asd เป็นฝ่ายชนะ

4. ผู้เล่น zxc เป็นฝ่ายแพ้แสดงผลการรันข้อความว่า “You Lose”



รูปที่ 4.8 แสดงผล zxc เป็นฝ่ายแพ้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5

บทสรุปและวิจารณ์

5.1 บทสรุปและวิจารณ์

ปัญหาการ Synchronize ระหว่าง applet กับ servlet คือใน applet นั้นจะมี thread คอยทำการ refresh update ข้อมูลให้สัมพันธ์กับ server เสมอ ซึ่งในบางครั้ง server ไม่สามารถตอบสนองให้ทันที่ได้ ทำให้ client applet นั้นส่ง request นั้นๆ ซ้ำกลับมายัง server ทำให้ server ต้องประมวลผล request นั้นแล้วส่งกลับไปยัง client อีก

ปัญหาการนำภาพขึ้นแสดงผล คือ applet ไม่สามารถแสดงรูปภาพทันทีแม้แต่ใช้ media tracker ช่วย แต่ก็ตาม ซึ่งมันเป็น กลไกการแสดงผลของ ภาษา JAVA

5.2 แนวทางในการพัฒนาต่อ

1. ปรับปรุงการส่งข้อมูลระหว่าง client กับ server ให้สามารถทำงานได้มีประสิทธิภาพมากขึ้น คือให้สามารถ synchronize ระหว่าง client กับ server ได้อย่างสมบูรณ์ โดยใช้ sequence number มากำกับแต่ละ request ที่ส่งให้กับ server ถ้า sequence อันไหนเคยทำแล้วก็ไม่ต้อง process ภายในซ้ำ แค่ส่งผลลัพธ์ กลับแค่นั้นพอ

2. เพิ่ม condition การสร้าง code script ให้มีมากขึ้น เช่น for loop , while loop เป็นต้น
3. เพิ่มส่วนที่ให้ ผู้เล่น สามารถเขียน code ลงได้เองคล้ายๆ กับ IDE โดยจะต้องมีส่วนของตัวแปรภาษาด้วย
4. ทำให้เกมให้มีการแสดงผลเป็นลักษณะ 3D

ภาคผนวก ก

คู่มือการติดตั้งโปรแกรม

วิธีการติดตั้ง

สำหรับผู้ที่มีโปรแกรม JRobot Battle programming online ไปใช้งานต้องทำตามคู่มือนี้เพื่อที่จะให้สามารถนำโปรแกรมนี้ไปใช้งานได้โดยมีประสิทธิภาพสูงสุด โดยต้องทำตามขั้นตอนดังนี้

ก.1 ติดตั้ง Web Server

โปรแกรมที่ต้องใช้

- Jakarta Tomcat 4.1.x or higher
- MySQL 4.1
- Mysql Connector 3.0.15 (Mysql Driver JDBC)
- JAVA J2SE 1.4.2.x or higher Runtime edition
- โปรแกรม JRobot battle programming online (library part)
- โปรแกรม JRobot battle programming online (web part)

ก.1.1 ทำการลงโปรแกรม Jakarta Tomcat 4.1.x ไว้ใน directory C:\jakarta-tomcat-4.1.31 และทำการเซต Environment variables 2 ค่าดังนี้

Variable name	Variable value
CATALINA_HOME	C:\jakarta-tomcat-4.1.31
CLASSPATH	C:\jakarta-tomcat-4.1.31\common\lib\servlet.jar C:\jakarta-tomcat-4.1.31\common\lib\mysql-connector-java-3.0.15-ga-bin.jar

ก.1.2 ทำการลงโปรแกรม MySQL 4.1

ก.1.3 ทำการลงโปรแกรม JAVA J2SE 1.4.2 ไว้ใน directory C:\j2sdk1.4.2 และทำการเซต Environment variables ดังนี้

Variable name	Variable value
JAVA_HOME	C:\j2sdk1.4.2
PATH	%JAVA_HOME%\bin C:\j2sdk1.4.2\bin

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

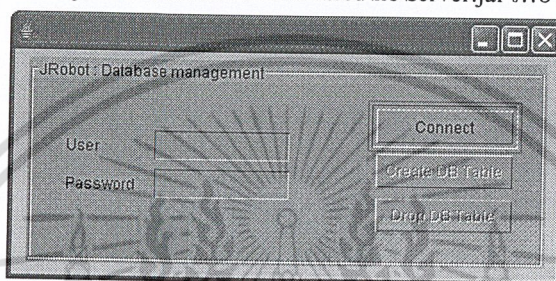
ก.1.4 นำ Mysql Connector 3.0.15 (Mysql Driver JDBC) ไปไว้ที่ C:\jakarta-tomcat-4.1.31\shared\lib

ก.1.5 นำโปรแกรม JRobot battle programming online (library part) ไปไว้ที่ C:\jakarta-tomcat-4.1.31\shared\lib

ก.1.6 นำโปรแกรม JRobot battle programming online (web part) ไปไว้ที่ C:\jakarta-tomcat-4.1.31\webapps\ROOT

ก.2 ทำการสร้างฐานข้อมูลที่ใช้ในโปรแกรม

ทำการเปิดโปรแกรม C:\jakarta-tomcat-4.1.31\shared\lib\server.jar เพื่อที่จะทำการสร้างฐานข้อมูล



รูปที่ ก.1 แสดงโปรแกรมที่ใช้ในการสร้างฐานข้อมูล

เมื่อทำการกรอก user และ password Admin ของ MySQL แล้วกดปุ่ม Connect เพื่อติดต่อกับฐานข้อมูลจากนั้นเราสามารถ กดปุ่ม Create DB Table เพื่อทำการ สร้าง Table ที่จำเป็นในโปรแกรม และปุ่ม Drop DB Table เพื่อทำการ Drop Table ที่ใช้ในโปรแกรม

ก.3 การ Start up Web Server

การ Start up web server นั้นเราจำเป็นต้องเปิด MySQL Database Service ก่อนโดยเรียกโปรแกรม winmysqldadmin ภายใต้ directory C:\MySql\bin จากนั้นไปทำการ Start up web server โดยเรียกโปรแกรม startup ภายใต้ directory C:\jakarta-tomcat-4.1.31\bin ดังรูป

```

C:\ Tomcat
1 01.T. 2548 23:23:52 org.apache.coyote.http11.Http11Protocol init
INFO: Initializing Coyote HTTP/1.1 on http-8080
Starting service Tomcat-Standalone
Apache Tomcat/4.1.31
1 01.T. 2548 23:24:05 org.apache.struts.util.PropertyMessageResources <init>
INFO: Initializing, config='org.apache.struts.util.LocalStrings', returnNull=true
1 01.T. 2548 23:24:05 org.apache.struts.util.PropertyMessageResources <init>
INFO: Initializing, config='org.apache.struts.action.ActionResources', returnNull=true
1 01.T. 2548 23:24:08 org.apache.struts.util.PropertyMessageResources <init>
INFO: Initializing, config='org.apache.webapp.admin.ApplicationResources', returnNull=true
1 01.T. 2548 23:24:15 org.apache.coyote.http11.Http11Protocol start
INFO: Starting Coyote HTTP/1.1 on http-8080
1 01.T. 2548 23:24:15 org.apache.jk.common.ChannelSocket init
INFO: JK2: ajp13 listening on /0.0.0.0:8009
1 01.T. 2548 23:24:15 org.apache.jk.server.JkMain start
INFO: Jk running ID=0 time=30/220 config=C:\jakarta-tomcat-4.1.31\conf\jk2.properties
  
```

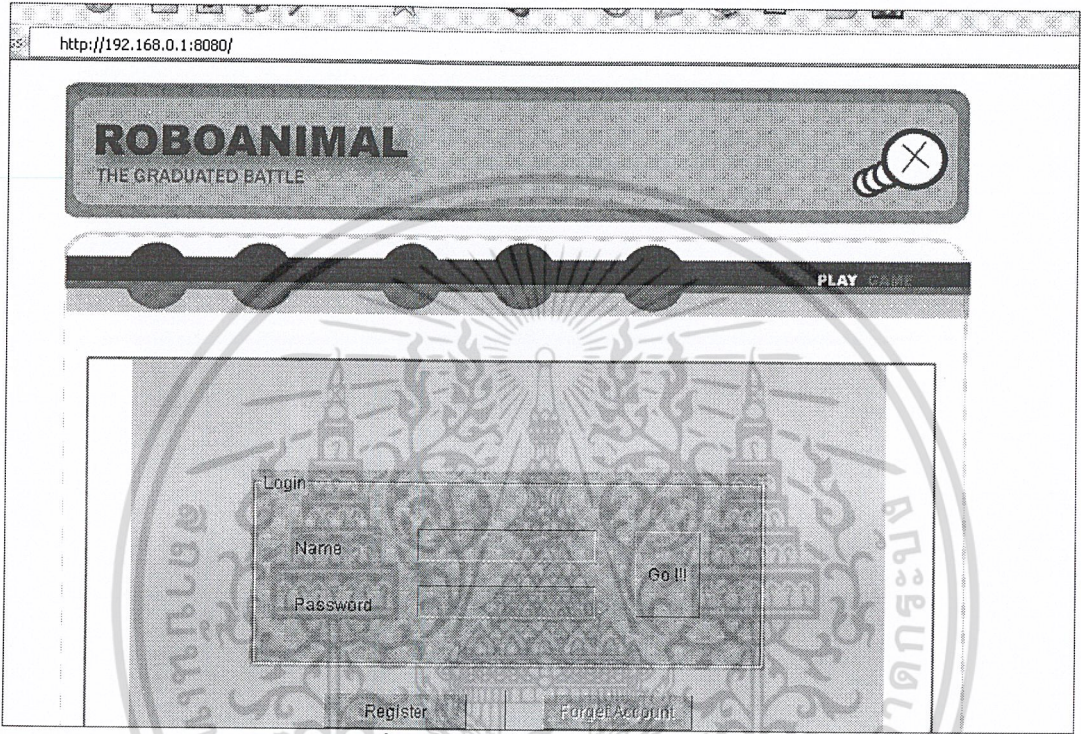
รูปที่ ก.2 แสดงโปรแกรม Tomcat ที่ใช้เป็น web server

ก.4 การ Shutdown Web Server

หลังจากที่ใช้งานโปรแกรม JRobot battle programming online เรียบร้อยแล้วหากต้องการจะ shutdown web server นั้นเราจะต้องไป stop service ของ MySQL ที่ System tray ก่อนจากนั้นไป Shutdown โปรแกรม Tomcat โดยไปเรียกโปรแกรม shutdown ภายใต้ directory C:\jakarta-tomcat-4.1.31\bin เพื่อให้ Tomcat ทำการ destroy Sevlet อย่างสมบูรณ์และผู้เล่นฝั่ง Client จะไม่สามารถติดต่อกับ web server ได้

ก.5 การเข้าเล่นเกม

การเล่นเกมนี้อาจจะเล่นผ่าน web browser เราสามารถใช้โปรแกรม browser เพื่อเข้ามาเล่นเกมนี้ได้แต่จะต้องติดตั้ง J2RE 1.4.2 or higher เพื่อที่จะให้สามารถ run java ได้โดยเข้าไปยัง url ต่อไปนี้ <http://192.168.0.1:8080/>



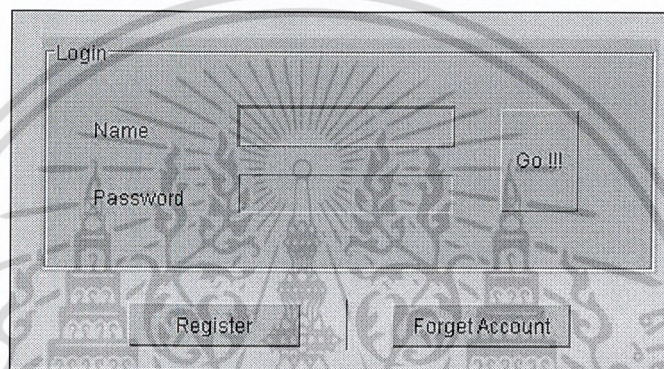
รูปที่ ก.3 แสดงฝั่งผู้เล่นที่เข้าไปเล่นเกม

ภาคผนวก ข

คู่มือการใช้งานโปรแกรม

วิธีการเล่นเกม

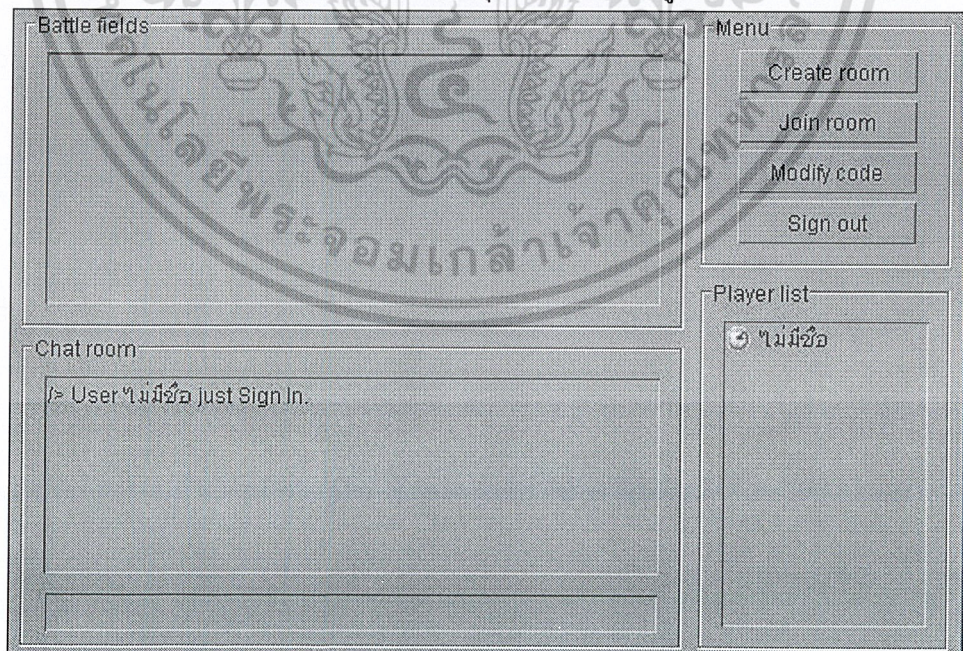
1. การเริ่มเข้าสู่เกมต้องทำการเข้าสู่ระบบ (Login) ด้วยการกรอก ชื่อผู้ใช้ (Name) และรหัสผ่าน (Password) เพื่อเข้าใช้งาน ซึ่งจะได้มาจากการสมัครสมาชิก (Register) ก่อน (ดูรายละเอียดได้ในหัวข้อวิธีสมัครสมาชิก)



The screenshot shows a login window titled 'Login'. It contains two input fields: 'Name' and 'Password'. To the right of the 'Name' field is a 'Go !!!' button. Below the 'Password' field are two buttons: 'Register' and 'Forget Account'.

รูปที่ ข.1 หน้าจอแสดงการกรอกชื่อและรหัสผ่าน

2. เมื่อกรอกชื่อและรหัสผ่านเรียบร้อยแล้วทำการกดปุ่ม GO!!! เพื่อเข้าสู่เกม



The screenshot shows the main game interface. It is divided into several sections: 'Battle fields' at the top left, 'Chat room' at the bottom left, 'Menu' at the top right, and 'Player list' at the bottom right. The 'Menu' section contains buttons for 'Create room', 'Join room', 'Modify code', and 'Sign out'. The 'Player list' section shows a single entry: 'ไม่มีชื่อ' (No name).

รูปที่ ข.2 ส่วนแสดงหน้าหลักของเกม

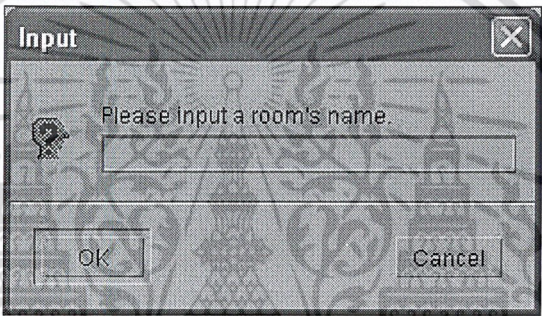
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในเมนูหลักสำหรับการเล่นเกมนั้นจะมีส่วนของการจัดการเกม 4 ส่วนด้วยกัน ซึ่งได้แก่

- Create Room
- Join Room
- Modify Code
- Sign Out

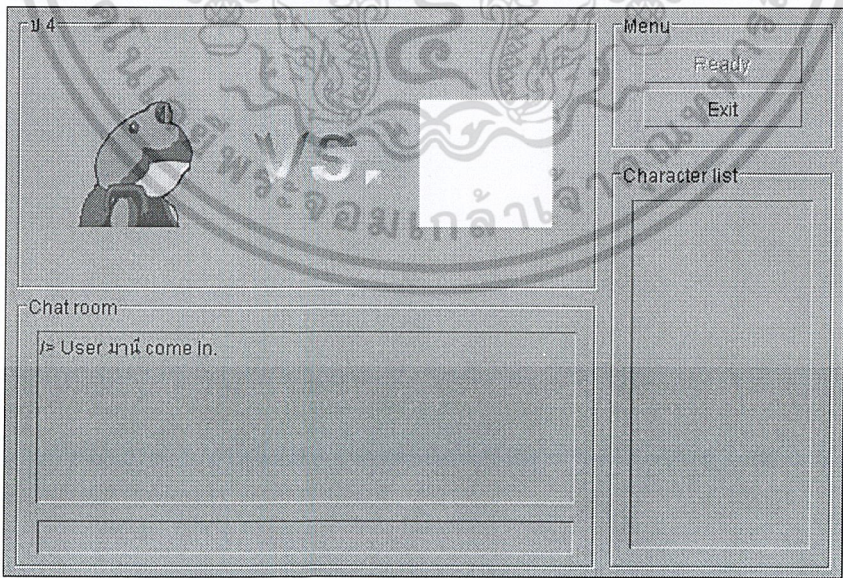
การสร้างห้อง (Create Room)

1. ในการเข้าเล่นเกมนั้น ผู้เล่นเกมสามารถเลือกได้ว่าจะทำการสร้างห้องขึ้นมาใหม่ ทำการคลิกที่ปุ่ม Create Room เมื่อเข้าไปแล้วระบบจะแสดงหน้าจอเพื่อให้ใส่ชื่อห้อง โดยเมื่อใส่ชื่อห้องที่ต้องการแล้วกด OK เพื่อยืนยันการสร้างห้อง หรือกด cancel เพื่อยกเลิกการสร้างห้อง



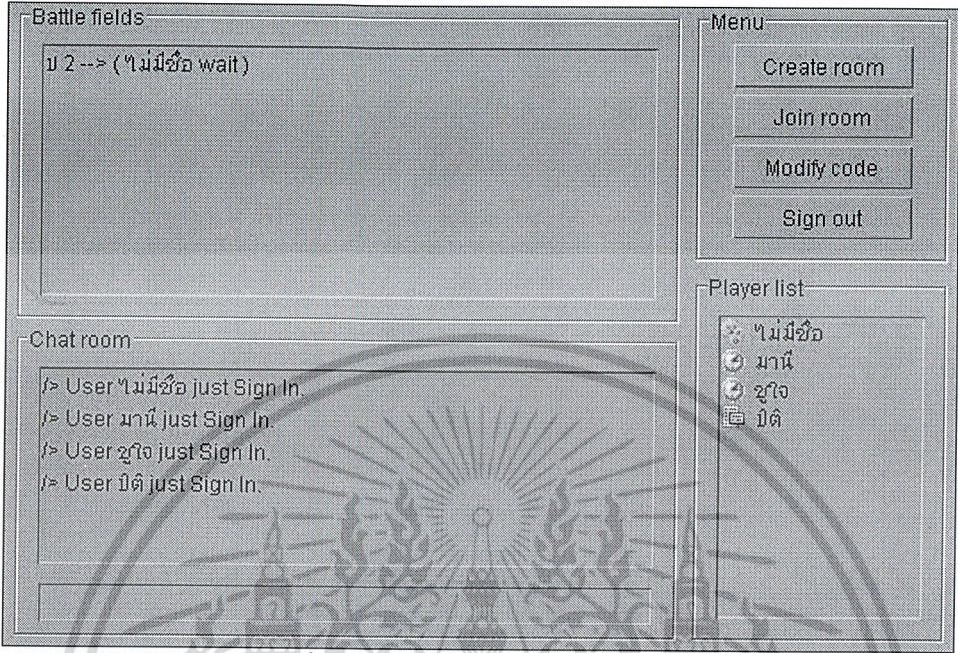
รูปที่ ข.3 ภาพหน้าจอแสดงการสร้างห้อง

2. เมื่อทำการสร้างห้องเรียบร้อยแล้วระบบจะแสดงหน้าจอแสดงห้องที่ได้สร้างขึ้น และผู้สร้างห้องรออีกฝ่ายหนึ่งเข้ามาเล่น ซึ่งอยู่ในสถานะรอคอย (wait)



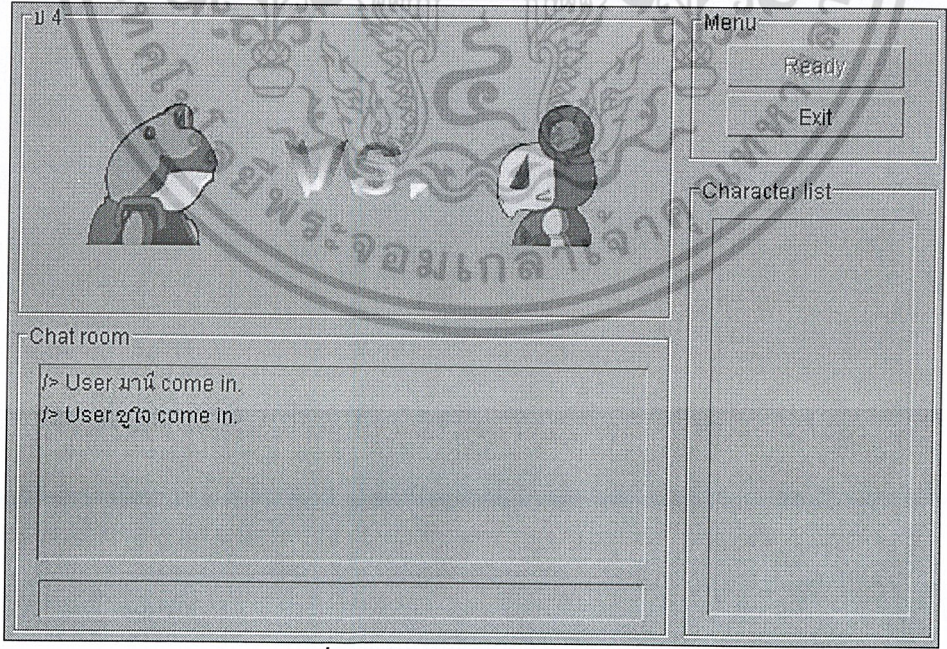
รูปที่ ข.4 หน้าจอที่จะเห็นเฉพาะผู้สร้างห้องเท่านั้น

ในส่วนของผู้เล่นนั้นจะเห็นห้องที่ได้ถูกสร้างไว้แล้วในช่องแสดงผล Battle fields



รูปที่ ข.5 หน้าจอที่จะเห็นโดยผู้เล่นทั่วไป

3. เมื่ออีกฝ่ายพร้อมเล่นแล้วจึงกดเข้ามาเพื่อเล่น และกดปุ่ม Ready



รูปที่ ข.6 หน้าจอแสดงการพร้อมการเล่นเกม

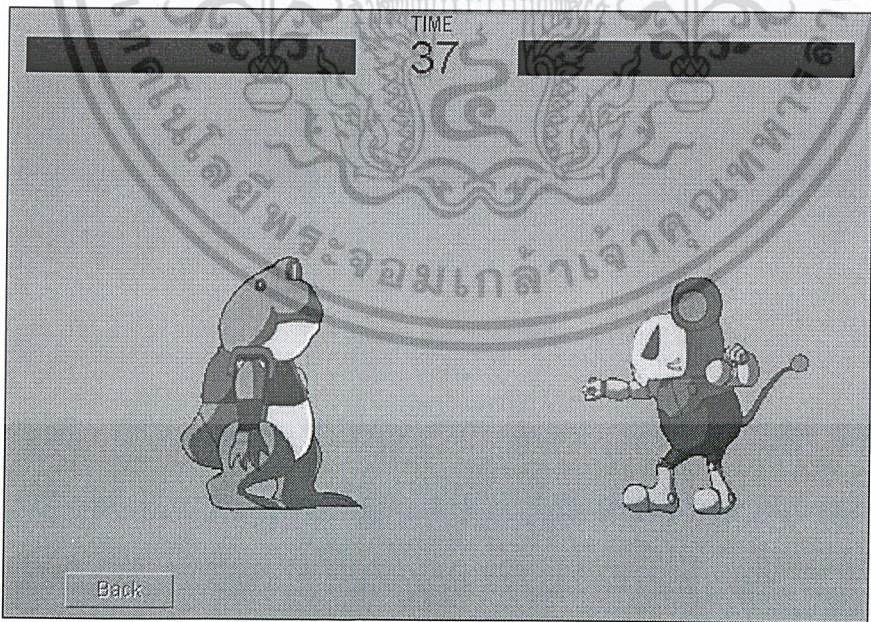
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4. เมื่อทำการกดปุ่ม Ready แล้วจะทำการ Load หน้าจอเกมขึ้นมา แสดงข้อความ “Loading”



รูปที่ ข.7 หน้าจอแสดงข้อความแจ้งเพื่อการโหลดเข้าสู่หน้าเกม

5. เมื่อระบบโหลดเรียบร้อยแล้วก็จะแสดงหน้าการเล่นเกมนขึ้นมา

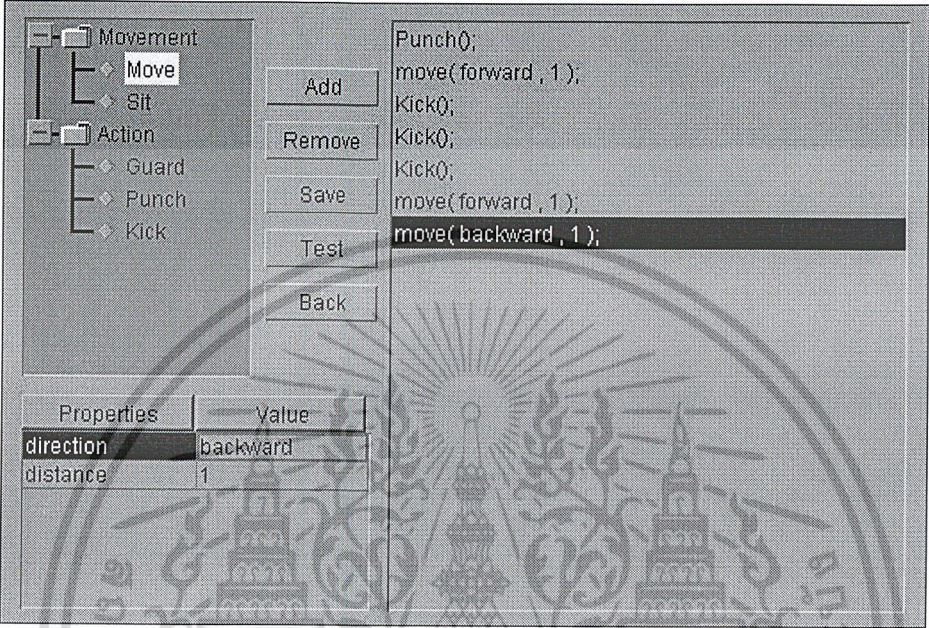


รูปที่ ข.8 หน้าจอแสดงผลการ Simulate เกม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การ Modify code

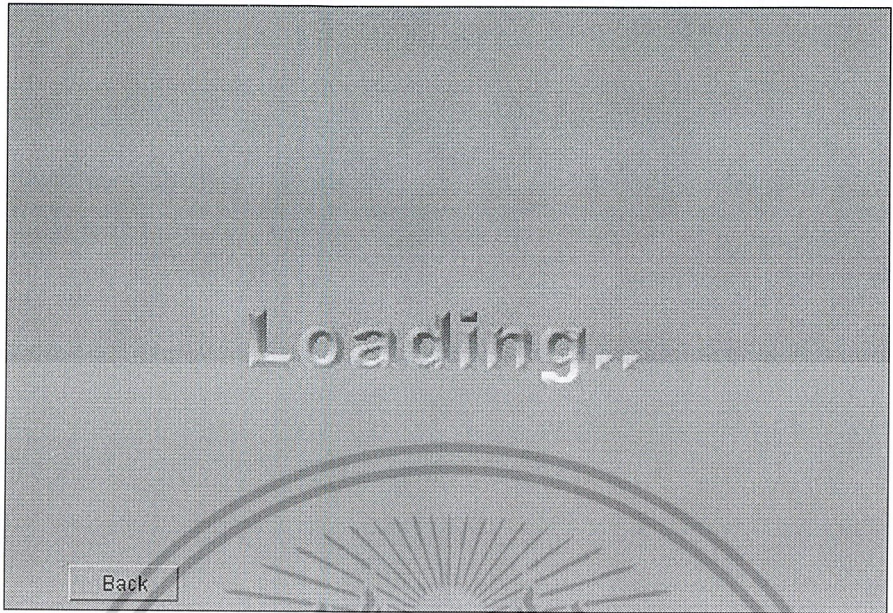
ในการกำหนดค่าต่างๆให้ตัวละครนั้นจะต้องกำหนดท่าทางการเคลื่อนไหว (Movement) และท่าทางการต่อสู้ (Action) ได้จากทางตัวเลือกด้านซ้ายที่ถูกแสดงไว้เป็นส่วนแบ่งแยกชัดเจน



รูปที่ ข.9 แสดงหน้าจอเพื่อเข้าไปทำการ *modify code*

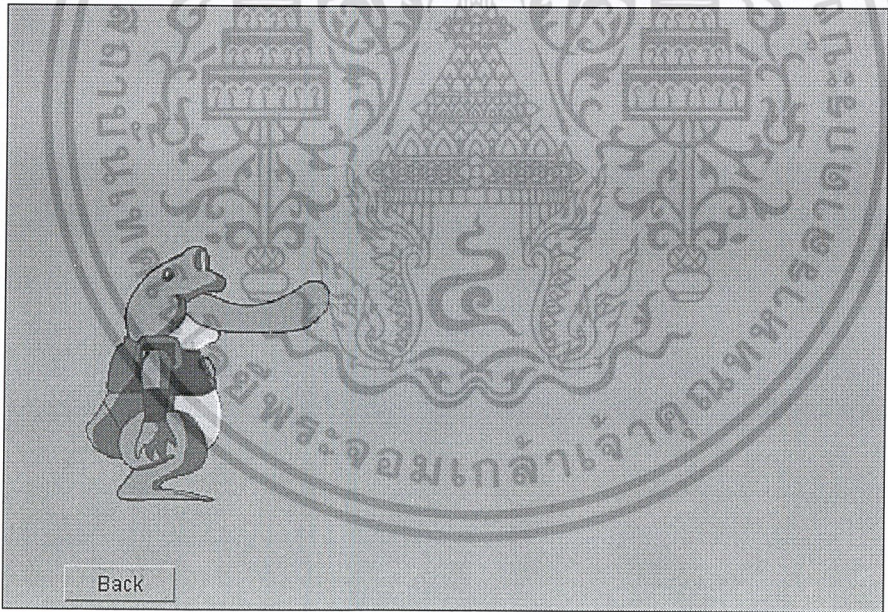
- ปุ่ม **Add** กดหลังจากที่ได้ทำการเลือกค่า Movement และ Action แล้ว เพื่อต้องการเพิ่ม script ให้กับตัวละคร
- ปุ่ม **Remove** กดเมื่อทำการเลือกลบ script ที่ไม่ต้องการออก
- ปุ่ม **Save** กดเมื่อต้องการบันทึกค่า Movement และ Action ซึ่งอาจจะบันทึกเก็บไว้หรือต้องการนำผลการปรับตั้งค่าไปรัน
- ปุ่ม **Test** กดเมื่อทำการกำหนดค่าแล้ว ซึ่งจะแสดงท่าทางที่ได้ปรับไปแล้ว จะทำการLoad เพื่อเข้าไปยังหน้าแสดงผลการรัน โดยหน้าจอจะขึ้นข้อความว่า Loading ดังรูปที่ ข.10

และตาราง Properties และ value เพื่อกำหนดในรายละเอียดของแต่ละท่าทาง



รูปที่ ข.10 หน้าจอแสดงข้อความแจ้งเพื่อการโหลดเข้าสู่หน้าเกม

หลังจากที่ได้ทำการ โหลดแล้ว จะเข้าสู่ระบบแล้วแสดงผลการทดสอบ



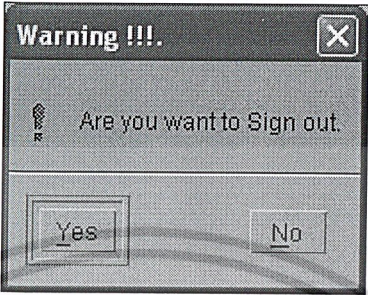
รูปที่ ข.11 หน้าจอแสดงผลการรันโปรแกรม

- ปุ่ม Back กดเมื่อต้องการกลับไปยังหน้าหลักของเกม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การ ออกจากระบบ (Sign out)

เมื่อต้องเลิกเล่นเกม และต้องการออกจากระบบแล้วให้ทำการคลิกเลือกที่ปุ่ม Sign out แล้วหน้าจอจะแสดงหน้าต่าง (Pop up) ขึ้นมาเพื่อยืนยันการออกจากระบบ

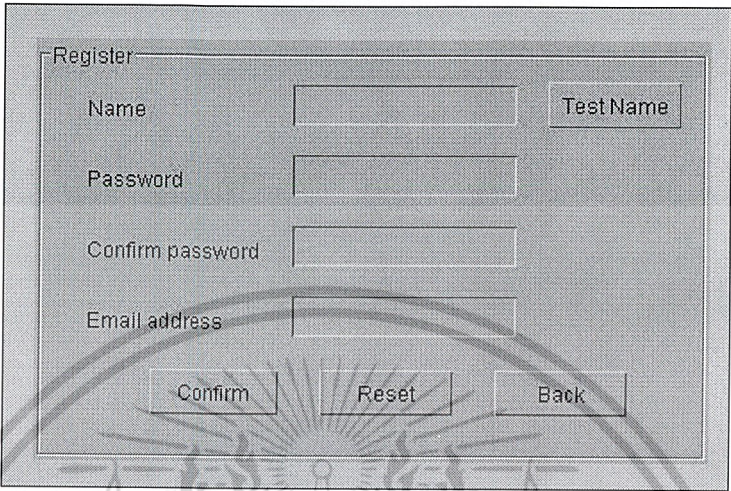


รูปที่ ข.12 POP UP ยืนยันการ Sign out

- กด Yes เพื่อยืนยันเลิกการเล่นเกม และออกจากระบบ
- กด No เพื่อสູ່หน้าเล่นเกม

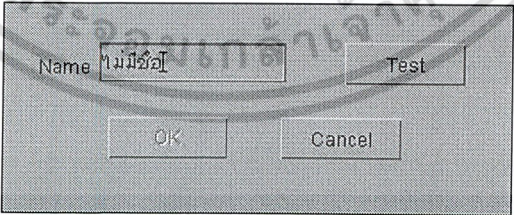
วิธีการลงทะเบียนเข้าเล่นเกม

1. การลงทะเบียน (Register) ต้องทำการกรอกข้อมูลชื่อ (Name), รหัสผ่าน(Password), ยืนยันรหัสผ่าน (Confirm Password) และอีเมลผู้ใช้ (Email address)



รูปที่ ข. 13 แสดงหน้าการสมัครสมาชิก

- กด **Confirm** เพื่อยืนยันการสมัครสมาชิก
 - กด **Reset** เพื่อเคลียร์ค่าข้อมูลสมัครสมาชิกให้เป็นค่าเริ่มต้น(Default Value)
 - กด **Back** เพื่อกลับไปยังหน้าหลัก
2. ในการสมัครสมาชิกนั้น เพื่อป้องกันการใช้ชื่อที่ซ้ำกัน จึงต้องทำการเลือกชื่อว่ามีอยู่ในระบบแล้วหรือไม่ โดยทำการทดสอบชื่อก่อนที่ปุ่ม **Test Name** จะทราบได้ทันทีว่าชื่อที่ต้องการสมัครนั้นซ้ำหรือไม่



รูปที่ ข.14 แสดงการทดสอบการเลือกชื่อ

ทำการกรอกชื่อที่ต้องการ กดที่ปุ่ม **Test** ระบบจะแสดงผลการทดสอบชื่อออกมา ซึ่งหากชื่อไม่ซ้ำก็จะสามารถกดปุ่ม **OK** เพื่อยืนยันการเลือกชื่อ หรือ **Cancel** เพื่อยกเลิกชื่อนั้น แต่หากชื่อนั้นมีผู้อื่นลงทะเบียนใช้งานแล้ว จะทำให้ไม่สามารถกดที่ปุ่ม **OK** ได้

ภาคผนวก ก.

Character ของตัวละคร

ก.1 แนวคิดในการสร้างตัวละคร

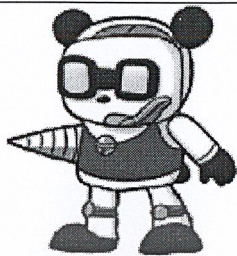
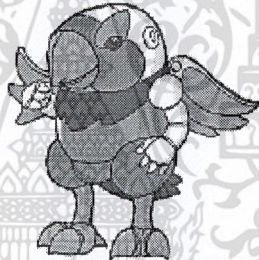
แนวคิดในการออกแบบตัวละครนั้น ได้สร้างคุณลักษณะเฉพาะหรือ Character ของตัวละครแต่ละตัวให้แตกต่างกันเพื่อความสนุกสนานของเกม โดยแนวคิดจากเหล่าบรรดาพวกสัตว์ทั้งหลายที่มารวมกันประลองฝีมือ ในนามของ Roboanimal จึงช่วยเพิ่มสีสันและความน่าสนใจให้เกมมากขึ้น



รูปที่ ก.1 แสดงภาพโปสเตอร์ของเกม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โดยได้ออกแบบตัวละครทั้งหมดออกมา 5 ตัว ดังนี้

ประเภทของสัตว์ : ชื่อในเกม	ภาพแสดง
1.หมีแพนด้า มีชื่อว่า “แพนดากิ (Pandaki)”	
2. ช้าง มีชื่อว่า “อิเลฟากิ (Elephaga)”	
3.นกแก้ว มีชื่อว่า “เพโทรริ (Petrori)”	
4. หนู มีชื่อว่า “ราบบิก้า (Rabbiga)”	
5. กบ มีชื่อว่า “ฟอร์คการิ (Forkari)”	

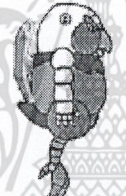
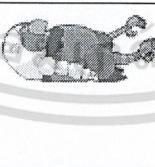
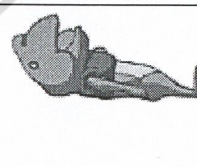
รูปที่ ค.2 แสดงภาพตัวละครทั้ง 5 ตัว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

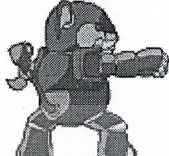
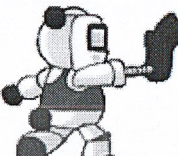
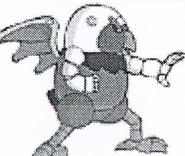
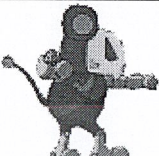
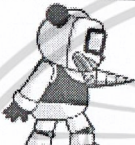
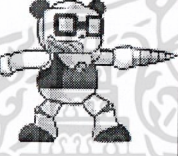
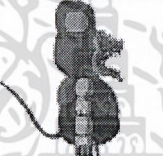
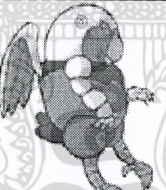
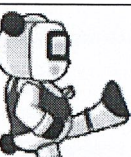
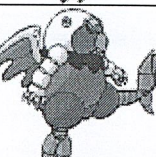
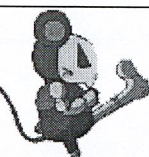
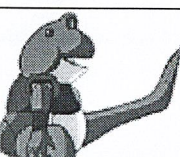
ค.2 แผนภาพแสดงท่าทางตัวละคร

ตารางที่ ค.1 และค.2 นั้นแสดงภาพท่าทางการเคลื่อนไหวและท่าทางการต่อสู้ของตัวละครซึ่งทำทางนั้นจะแสดงให้เห็นทั้งทางซ้ายและทางขวา และจะอธิบายแยกเป็นส่วนแรกของท่าทางการเคลื่อนไหวและส่วนที่สองนั้นเป็นท่าทางการต่อสู้

ด้านขวา (Right)

Elephant	Panda	Parrot	Rat	Frog
ส่วนที่ 1 ท่าทางการเคลื่อนไหว				
BASE				
				
WALK				
				
				
DEAD				
				
ส่วนที่ 2 ท่าทางการต่อสู้				
GUARD				
				

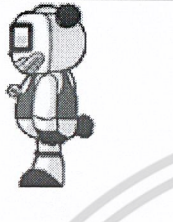
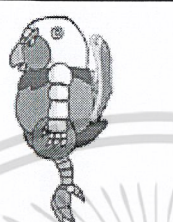
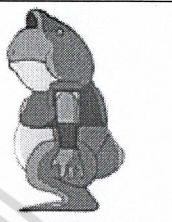
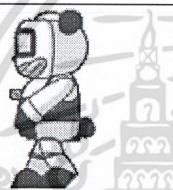
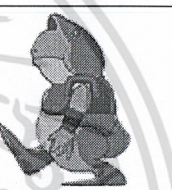
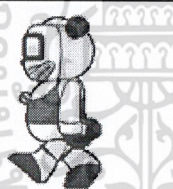
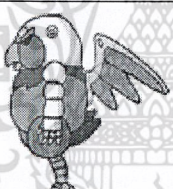
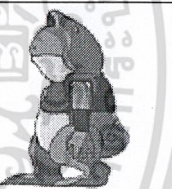
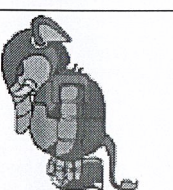
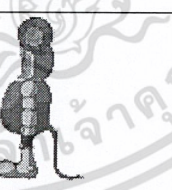
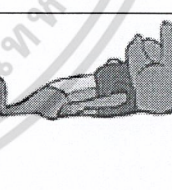
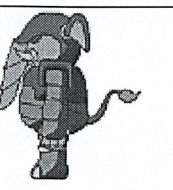
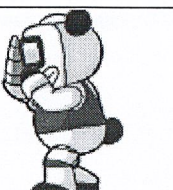
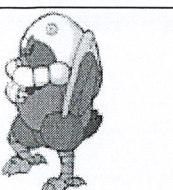
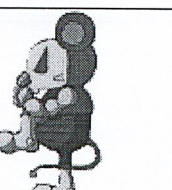
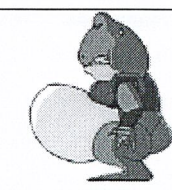
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

PUNCH				
				
				
SPECAIL				
				
				
DAMAGE				
				
KICK				
				
				

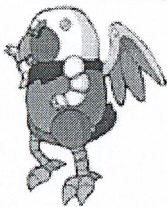
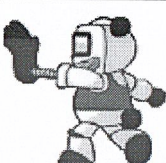
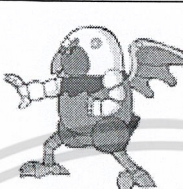
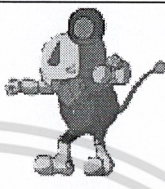
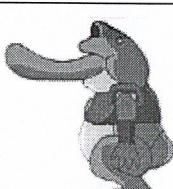
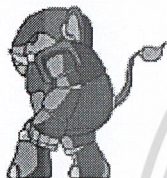
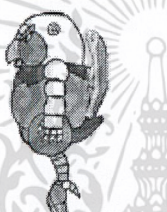
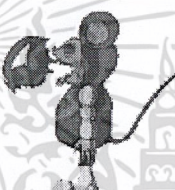
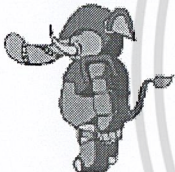
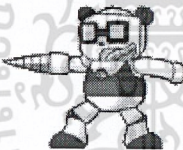
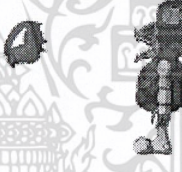
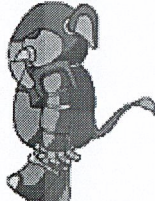
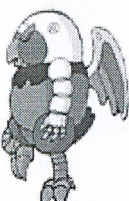
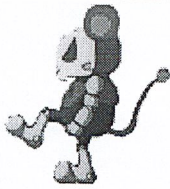
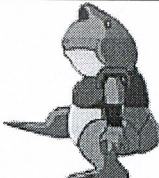
รูปที่ ค.3 แสดงท่าทางการเคลื่อนไหวและต่อสู้ของตัวละครทางด้านขวา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

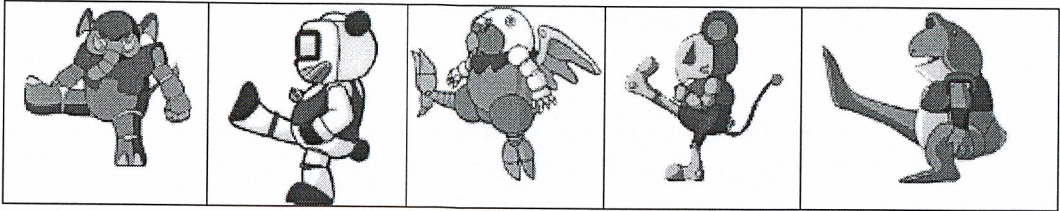
ด้านซ้าย (Left)

Elephant	Panda	Parrot	Rat	Frog
ส่วนที่ 1 ทำทางการเคลื่อนไหว				
BASE				
				
WALK				
				
				
DEAD				
				
ส่วนที่ 2 ทำทางการต่อสู้				
GUARD				
				

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

PUNCH				
				
PUNCH(ต่อ)				
				
SPECAIL				
				
				
DAMAGE				
				
KICK				
				

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ ค.4 แสดงท่าทางการเคลื่อนไหวและต่อสู้ของตัวละครทางด้านซ้าย



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บรรณานุกรม

- [1] H.M.Deitel ,P.J.Deitel. **Java How To Program 5 Edition**. Upper Saddle River, New Jersey:Prentice-Hall International,c2003
- [2] ดร.วีระศักดิ์ ชิงฉาวร. **Java Programming Volume I**. – กรุงเทพฯ:ซีเอ็ดดูเคชั่น,2545.
- [3] ดร.วีระศักดิ์ ชิงฉาวร. **Java Programming Volume II**. – กรุงเทพฯ:ซีเอ็ดดูเคชั่น,2546.
- [4] ดร.วีระศักดิ์ ชิงฉาวร. **Java Programming Volume III**. – กรุงเทพฯ:ซีเอ็ดดูเคชั่น,2547.
- [5] <http://java.sun.com/>
- [6] <http://www.eclipse.org/>
- [7] <http://www.javaworld.com/>
- [8] <http://jakarta.apache.org>
- [9] <http://www.jboss.org>

