# บทที่ 3

# เชลล์ในระบบยูนิกซ์

เชลล์ในระบบยูนิกซ์ (UNIX shell) หมายถึงโปรแกรมที่ระบบยูนิกซ์ใช้ติดต่อสื่อสารกับผู้ใช้ยูนิกซ์ เชลล์ที่ใช้ในระบบยูนิกซ์ทั่วไปคือ เบิร์นเชลล์และคอร์นเชลล์ สำหรับระบบยูนิกซ์ซิสเต็มไฟว์ และ ซีเชลล์ สำหรับระบบยูนิกซ์แบบเบิร์กเลย์ (Berkeley) (Thomas, and Farrow, 1989) ในบทนี้กล่าวถึงเฉพาะรายละเอียดของเบิร์นเชลล์

## ลักษณะการใช้งานเชลล์ในระบบยูนิกซ์

การใช้งานเชลล์ทำได้ 2 แบบ (Sebes, 1988)

1. ตัวแปลคำสั่งงาน (command interpreter) หมายถึงเชลล์ที่ทำงานในโมดโต้ตอบ เชลล์จะรับคำสั่งงานที่ผู้ใช้ป้อนเข้าไปทางแป้นพิมพ์ แปลความและปฏิบัติการ (execute) คำสั่งงานนั้น

2. ภาษาทำโปรแกรมคำสั่งงาน (command programming language) โดยการสร้างแฟ้มข้อความที่เรียกว่า ชุดคำสั่งเชลล์ (shell script) ซึ่งต่อไปจะเรียกว่า โปรแกรม จากนั้นจึงนำมาประมวลผลโดยใช้เชลล์

## องค์ประกอบทางภาษาของภาษาทำโปรแกรมเชลล์

ภาษาทำโปรแกรมเชลล์ ประกอบด้วยองค์ประกอบทางภาษาต่างๆดังต่อไปนี้

1. หมายเหตุ

เป็นข้อความที่เชลล์ไม่ประมวลผล ผู้เขียนโปรแกรมสามารถใช้หมายเหตุเพื่ออธิบายการทำงานของโปรแกรม เพิ่มเติมข้อแนะนำ หรือข้อความอื่นลงในโปรแกรมได้โดยไม่มีผลต่อการทำงานของโปรแกรม

2. คำสั่ง

คำสั่งของเชลล์ มีลักษณะเป็นข้อความที่ประกอบด้วยคำตั้งแต่หนึ่งคำขึ้นไป คำแรกแสดงชื่อคำสั่ง คำอื่นๆเป็นข้อมูลที่คำสั่งใช้ในการประมวลผล เรียกว่า อาร์กิวเมนต์ (argument) คำสั่งที่ใช้งานได้ในเชลล์มีอยู่ 2 ประเภท

2.1 คำสั่งภายใน หมายถึงคำสั่งที่เชลล์สามารถทำการประมวลผลได้เอง เช่น ซีดี (cd) เอ็คโค (echo) เอ็กซิต (exit) รีด (read)

2.2 คำสั่งภายนอก หมายถึงคำสั่งที่เชลล์ไม่สามารถประมวลผลเอง แต่จะเรียกใช้โปรแกรมที่มีชื่อตรงกับชื่อคำสั่งมาประมวลผล เช่น แค็ท (cat) ฮู (who) โปรแกรมเหล่านี้เรียกอีกอย่างว่า โปรแกรมอรรถประโยชน์ (utility program)

3. การควบคุมสายงาน (flow control)

3.1 การทำแบบมีเงื่อนไข (if) เป็นการเลือกทำคำสั่งที่กำหนดให้หนึ่งหรือสองชุดโดยตัดสินจากค่าความจริงของเงื่อนไขที่กำหนดให้

3.2 การวนซ้ำ เป็นการทำคำสั่งที่กำหนดให้ แบ่งได้เป็น 3 แบบ

3.2.1 วนซ้ำสำหรับรายการข้อมูลที่กำหนดให้ (for)

3.2.2 วนซ้ำจนเงื่อนไขที่กำหนดให้เป็นจริง (until)

3.2.3 วนซ้ำจนเงื่อนไขที่กำหนดให้เป็นเท็จ (while)

3.3 การเลือกทำโดยเงื่อนไข (case) เป็นการเลือกทำคำสั่งที่กำหนดให้หนึ่งชุดหรือมากกว่า โดยตัดสินจากค่าของข้อมูลที่กำหนดให้

4. การเบี่ยงเบนอินพุตและเอาต์พุต (I/O redirection)

โดยปกติ กระบวนการในระบบยูนิกซ์จะมีอินพุตและเอาต์พุตเชื่อมอยู่กับแป้นพิมพ์และจอภาพที่เทอร์มินอลของผู้ใช้ตามลำดับ การเบี่ยงเบนอินพุตและเอาต์พุตเป็นการกำหนดให้อินพุตหรือเอาต์พุตของกระบวนการเชื่อมกับแฟ้มข้อมูลที่ต้องการได้

5. ตัวแปรเชลล์ (shell variable)

มีหน้าที่เช่นเดียวกันกับตัวแปรในภาษาทำโปรแกรมทั่วไป

6. การทำงานเบื้องหลัง (background job)

เป็นการกำหนดให้การประมวลผลของงานเกิดขึ้นเบื้องหลัง ซึ่งหมายถึงอินพุตมาตรฐานของงานไม่ได้เชื่อมกับอินพุตมาตรฐานของเชลล์ และเชลล์จะประมวลผลคำสั่งถัดไปได้เลยโดยไม่ต้องรอให้งานนั้นเสร็จสิ้นเสียก่อน

7. ไปป์ไลน์ (pipeline)

เป็นชุดของคำสั่งที่เชื่อมกันด้วยไปป์ (pipe) ไปป์เป็นการเชื่อมเอาต์พุตของกระบวนการหนึ่งเข้ากับอินพุตของอีกกระบวนการหนึ่ง

8. ฟังก์ชันเชลล์ (shell function)

เป็นชุดคำสั่งเชลล์เช่นเดียวกับการสร้างโปรแกรม แต่การประมวลผลโปรแกรมเชลล์ทำโดยกระบวนการที่ถูกสร้างขึ้นใหม่ ในขณะที่การประมวลผลฟังก์ชันเชลล์จะทำโดยเชลล์ที่เรียกใช้ฟังก์ชัน จึงทำให้ผลลัพธ์ของการประมวลผลฟังก์ชันเชลล์มีผลต่อเชลล์ที่เรียกฟังก์ชันด้วย การเลือกพัฒนากระบวนความหนึ่งขึ้นเป็นฟังก์ชันเชลล์ หรือเป็นโปรแกรมเชลล์ จะต้องคำนึงถึงข้อแตกต่างนี้ด้วย

รีสตริกเชลล์ (restricted shell)

รีสตริกเชลล์ เป็นเชลล์ที่มีการจำกัดการใช้งานของผู้ใช้ยูนิกซ์ เพื่อที่จะควบคุมความปลอดภัยของระบบยูนิกซ์ เนื่องจากว่า มีบางกรณีที่การให้ผู้ใช้ยูนิกซ์ได้ใช้งานเชลล์จะทำให้เสี่ยงต่อความปลอดภัยต่อระบบยูนิกซ์ รีสตริกเชลล์มีคุณสมบัติและการใช้งานเหมือนกับเชลล์ในระบบยูนิกซ์ เพียงแต่มีข้อจำกัดเพิ่มเติมในการใช้งานดังต่อไปนี้

1. ไม่ให้ย้ายไดเรกทอรีที่กำลังใช้งาน (working directory)
2. ไม่ให้ตั้งค่าของตัวแปรชื่อพาธ (PATH) และ เชลล์ (SHELL)
3. ไม่ให้อ้างถึงชื่อคำสั่งงานที่มีเครื่องหมายแสลช (/) อยู่ด้วย

4. ไม่ให้เบี่ยงเบนเอาต์พุต (> และ >>)

ข้อจำกัดข้างต้นจะมีผลต่อการใช้งานแบบโต้ตอบเท่านั้น นอกจากนี้ ในกรณีที่เป็นเชลล์หลังการลงบันทึกเข้า (login shell) ซึ่งจะมีการประมวลผลคำสั่งในแฟ้มข้อมูลชื่อ ".โพรไฟล์" (.profile) ก่อน ข้อจำกัดข้างต้นจะเริ่มมีผลหลังจากที่เชลล์ประมวลผลคำสั่งในแฟ้มดังกล่าวแล้วเสร็จ

<u>เปรียบเทียบเชลล์แบบต่างๆ</u>

Sebes ได้ทำการเปรียบเทียบเบิร์นเชลล์ซีเชลล์และคอร์นเชลล์ไว้ สรุปได้ดังนี้

1. ในการทำงานแบบโต้ตอบนั้น ซีเชลล์และคอร์นเชลล์มีความสามารถเหนือกว่าเบิร์นเชลล์ เช่น สามารถเก็บประวัติการใช้คำสั่ง (history) เพื่อการใช้งานในภายหลัง สามารถสร้างคำเหมือน (alias)

2. ในการใช้งานเป็นภาษาทำโปรแกรม เบิร์นเชลล์และคอร์นเชลล์มีความสามารถเหนือกว่าซีเชลล์ เช่น เบี่ยงเบนอินพุตและเอาต์พุตได้คล่องตัวกว่า สามารถดักสัญญาณ (signal trapping) ชนิดต่างๆได้คล่องตัวกว่า

3. เบิร์นเชลล์ขาดความสามารถในด้านการคำนวณ

4. เบิร์นเชลล์สามารถแปลความคำสั่งงานได้รวดเร็วที่สุด เนื่องจากการที่มีชุดของคำสั่งที่เล็กที่สุดและซับซ้อนน้อยที่สุด

5. คอร์นเชลล์เป็นเชลล์ที่ได้รับการพัฒนาขึ้นหลังสุด ในปัจจุบัน จึงมีใช้งานไม่แพร่หลายเหมือนกับเบิร์นเชลล์และซีเชลล์