**บทที่ 2**

**ทฤษฎีที่เกี่ยวข้อง**

**2.1 การสร้างและพัฒนาเว็บ**

การเขียนและพัฒนาเว็บไซต์ มีองค์ความรู้มากมายที่เราต้องการทราบ แต่โดยพื้นฐานหลักๆ ของการพัฒนาเว็บแอพพลิเคชันจะประกอบไปด้วย 5 ส่วนสำคัญ ดังนี้

1) HTML สำหรับใช้เป็นโครงสร้างของเว็บเพจ

2.) CSS ใช้ตกแต่งหน้าตาของเว็บเพจ

3.) Client Side Script คือ สคริปต์ที่ใช้กำหนดคุณสมบัติเอลิเมนต์ต่างๆ ของเว็บเพจ และสามารถกำหนดลูกเล่นต่างๆ เพื่อโต้ตอบกับผู้ใช้ ตัวอย่าง Client Side Script ที่นิยมใช้งานในปัจจุบัน ได้แก่ JavaScript

4.) Server Side Script เป็นสคริปต์ที่จะแปลความหมายที่ฝั่งเซิร์ฟเวอร์ (Server) ใช้เพื่อติดต่อ, ส่งข้อมูล, เขียนข้อมูลไปยังเซิร์ฟเวอร์ (Server) เช่น เขียนสคริปต์เพื่อส่งรหัสนักเรียนไปยังเซิร์ฟเวอร์ จากนั้นเซิร์ฟเวอร์ก็ส่งรายละเอียดของนักเรียนคนนั้นกลับมาแสดงบนเว็บเว็บเพจ เป็นต้น Server Side Script ที่นิยมใช้กันในปัจจุบันได้แก่ PHP, ASP, Python, และ Ryby เป็นต้น

5) ฐานข้อมูล (Database) คือ ส่วนที่ใช้เก็บข้อมูล ซึ่งมีโครงสร้างที่แน่นอน และถูกจัดการอย่างเป็นระบบ เราสามารถเขียนข้อมูลลงในฐานข้อมูล หรือดึงข้อมูลที่ต้องการมาใช้กับเว็บไวต์ผ่านทาง Server Side Script (เช่น PHP, Python) โดยฐานข้อมูลที่นิยมใช้งาน ได้แก่ MySQL, Oracle และ Microsolf SQL Server เป็นต้น

**2.1.1 ภาษาเอชทีเอ็มแอล (HTML)**

HTML ย่อมาจากคำว่า Hypertext Markup Language เป็นภาษาคอมพิวเตอร์ที่ใช้สำหรับสร้างเว็บเพจ มีนามสกุลไฟล์เป็น .html ข้อความที่อยู่ในไฟล์ HTML ไม่ใช่ข้อความธรรมดาแต่มีโครงสร้างและรุปแบบที่แน่นอน โดยจะใช้แท็ก (Tag) เป็นตัวกำหนดรูปแบบการแสดงผล

HTML เป็นโค้ดที่มีโครงสร้าง ดังนั้นก่อนที่ผู้ใช้ทั่วไปจะนำเอา HTML ไปใช้งาน จะต้องแปลงโค้ด HTML ให้กลายเป็นเว็บเพจเสียก่อน ซึ่งซอร์ฟแวร์ที่ใช้สำหรับแปลงโค้ด HTML ให้กลายเป็น เว็บเพจ เรียกว่า เว็บเบราว์เซอร์ หรือเรียกย่อๆ ว่า เบราว์เซอร์ (Web Browser หรือ Browser)

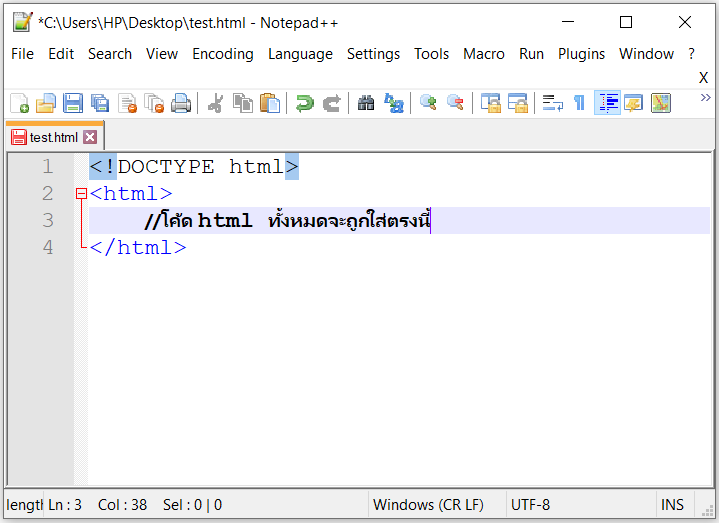
เบราว์เซอร์จะต้องถูกติดตั้งที่เครื่องคอมพิวเตอร์ของผู้ใช้ (Client) ทำหน้าที่แปลงโค้ด HTML, CSS, และตีความหมายโค้ด LavaScript เพื่อแสดงผลลัพธ์เป็นเว็บเพจต่อไป ในปัจจุบันเว็บเบราว์เซอร์ที่นิยมแพร่หลาย ได้แก่ Internet Explorer, Microsoft Edge, Firefox, Safari, Google Chrome และ Opera เป็นต้น

**2.1.1.1 เอลิเมนต์ (Element) และการใช้แท็ก (Tag)**

เอลิเมนต์ (Element) คือส่วนประกอบที่เป็นหน่วยย่อยของโครงสร้าง HTML ในแต่ละเอลิเมนต์จะใช้แท็ก (Tag) เพื่อบอกให้ทราบว่าแต่ละเอลิเมนต์มีหน้าที่และมีโครงสร้างเป็นเช่นไร

แท็กแรกที่จะกล่าวถึงคือ แท็ก <html> เป็นแท็กที่ใช้บอกขอบเขตของโค้ด HTML เป็นการประกาศให้ทราบว่าข้อความทั้งหมดที่อยู่ภายในแท็กนี้คือโค้ด html ซึ่งเบราว์เซอร์จะทำหน้าที่แปลความหมายของโค้ดนี้เพื่อแสดงผลเป็นเว็บเพจต่อไป

เอลิเมนต์ html จะประกอบไปด้วย แท็กเปิด (start tag) เพื่อบอกจุดเริ่มต้น และแท็กปิด (end tag) เพื่อบอกจุดสิ้นสุดของเอลิเมนต์ ดังนี้

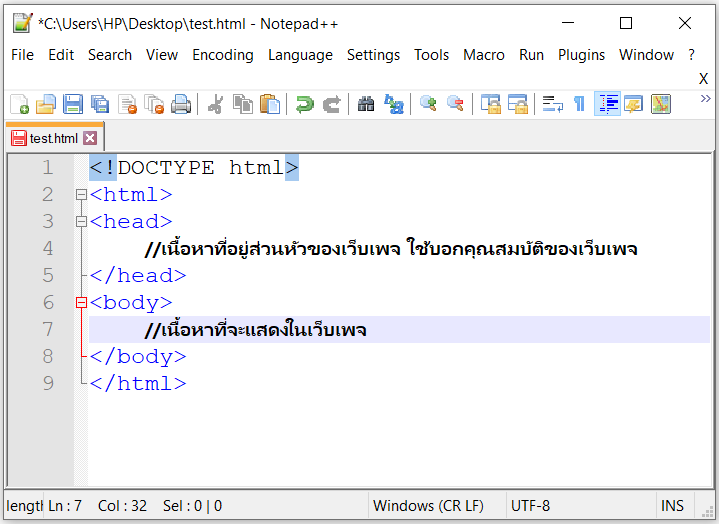


**รูปที่** **2.1 แท็ก html**

จากตัวอย่างโค้ดด้านบน <html> คือโค้ดที่เป้นแท็กเปิด เพื่อบอกจุดเริ่มต้นของเอลิเมนต์ ส่วนโค้ด</html> คือแท็กปิดซึ่งใช้บอกจุดสิ้นสุดของเอลิเมนต์

**2.1.1.2 รูปแบบมาตรฐานของ HTML**

ในเว็บเพจจะประกอบไปด้วย 3 ส่วนหลักๆ คือ ส่วนประกาศ ส่วนหัวของเว็บเพจ และส่วนของเนื้อหาของเว็บเพจ โดยใช้แท็ก <html>, <head> และ <body> ตามลำดับ



**รูปที่** **2.2 แบบมาตรฐานของ html**

จากโค้ดตัวอย่างด้านบน เราสามารถแบ่งโค้ดออกเป็น 3 ส่วน ดังนี้

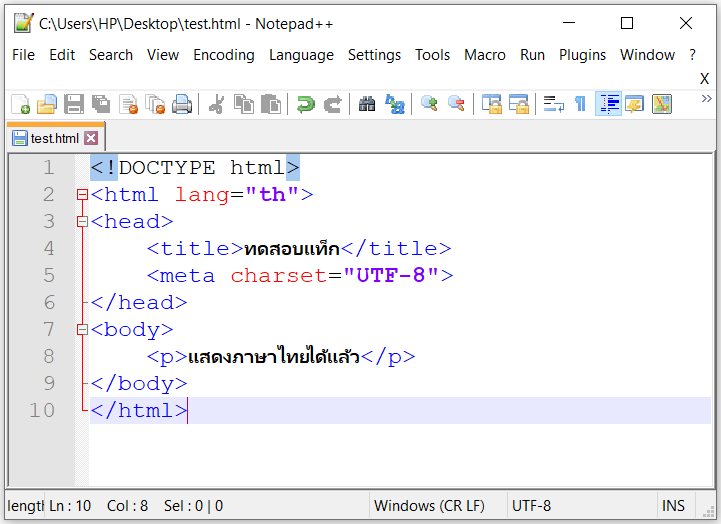
ส่วนที่ 1 คือแท็ก <html> ที่อยู่ในบรรทัดสองตามตัวอย่างนั้นใช้เพื่อประกาศว่า เนื้อหาที่อยู่ในภายในแท็กคือโค้ดที่จะให้เบราเซอร์ตีความเพื่อแสดงผลบนเบราว์เซอร์

ส่วนที่ 2 คือแท็ก <head> เป็นแท็กที่ใช้กับคุณสมบัติของเว็บเพจ เช่น กำหนดชื่อเพจ (title) หรือกำหนดสไต์ (style) ให้กับเว็บเพจ เป็นต้น โดยแท็ก <head> จะอยู่ภายใต้แท็ก <html>

ส่วนที่ 3 คือแท็ก <body> เป็นแท็กที่ใช้กับเนื้อหาของเว็บเพจ โดยแท็ก <body> นี้จะอยู่ภายใต้แท็ก <html> เช่นเดียวกัน

**2.1.1.3 ภาษาไทยใน HTML**

หากต้องการให้เว็บเพจสามารถแสดงภาษาไทยได้ เราจะใส่แท็ก <meta charset=”UTF-8> เข้าไปใส่ไว้ในแท็ก <head> เพื่อบอกให้เบราว์เซอร์ทราบว่า เว็บเพจนี้ใช้ชุดตัวอักษร (character set) ตามมาตรฐาน UTF-8 ซึ่งเป็นตัวอักษรแบบ Unicode ที่รองรับภาษาไทยเรียบร้อยแล้ว



**รูปที่** **2.3 การกำหนดใช้ภาษาไทยในเว็บเพจ**

จากโค้ดตามตัวอย่างข้างบน จะมีโค้ดส่วนที่เราต้องให้ความสนใจอยู่ 2 ส่วน ดังนี้

ส่วนที่ 1 คำสั่ง <html lang=”th”> หมายถึง ต้องการให้เนื้อหาที่อยู่ภายในเอลิเมนต์นี้เป็นภาษาไทย เราสามารถกำหนดภาษาอื่นๆ ได้ตามความเหมาะสม เช่น ถ้าต้องการให้เว็บเพจมีเนื้อหาเป็นภาษาฝรั่งเศษ ก็กำหนดเป็น < html lang=”fr”> เป็นต้น

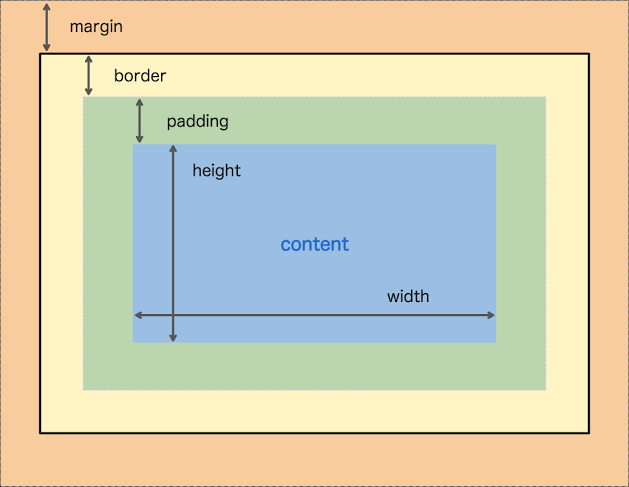
ส่วนที่ 2 คำสั่ง <meta charset=”UTF-8”> หมายถึง การเลือกชุดตัวอักษรที่ใช้แสดงผลบนเว็บเพจ โดยเราได้เลือกชุดตัวอักษรแบบ UTF-8 ซึ่งเป็นตัวอักษร Unicode ที่รองรับภาษาต่างๆ รวมทั้งภาษาไทย

**2.1.2 ภาษาซีเอสเอส (CSS)**

CSS ย่อมาจากคำว่า Cascading Style Sheet ซึ่งเราจะใช้ CSS เพื่อแก้ไขคุณสมบัติของเว็บเพจ เพื่อให้มีหน้าตา สีสัน หรือรูปแบบเป็นไปตามที่ต้องการ CSS มีลักษณะคล้ายหน้ากากสำหรับเปลี่ยนรูปร่างหน้าตาของเว็บเพจให้มีรูปแบบที่ต่างออกไป เช่น การเปลี่ยนสีสัน การจัดวางข้อความ รูปแบบตัวอักษรที่ใช้ ตำแหน่งของรูปภาพ ระยะห่างจากขอบเว็บเพจ ฯลฯ สรุปก็คือ เว็บเพจเดียวกัน หากมีการใช้ CSS ที่ต่างกัน รูปร่างหน้าตาก็จะมีผลลัพธ์ที่ต่างออกไป

**2.1.2.1 CSS box model**

ในทุกๆ เอลิเมนต์ (element) บนเว็บเพจ จะมีโครงสร้างหนึ่งที่เหมือนกันคือ box model โดยมีลักษณะคล้ายกล่องสี่เหลี่ยม ที่มีคุณสมบัติ (properties) พื้นฐานแบบเดียวกัน ได้แก่ margin, padding, width และ height โดยค่าเหล่านี้จะอ้างอิงกรอบ (border) และข้อความที่อยู่ภายในเอลิเมนต์ ดังรูป



**รูปที่** **2.4 CSS box model**

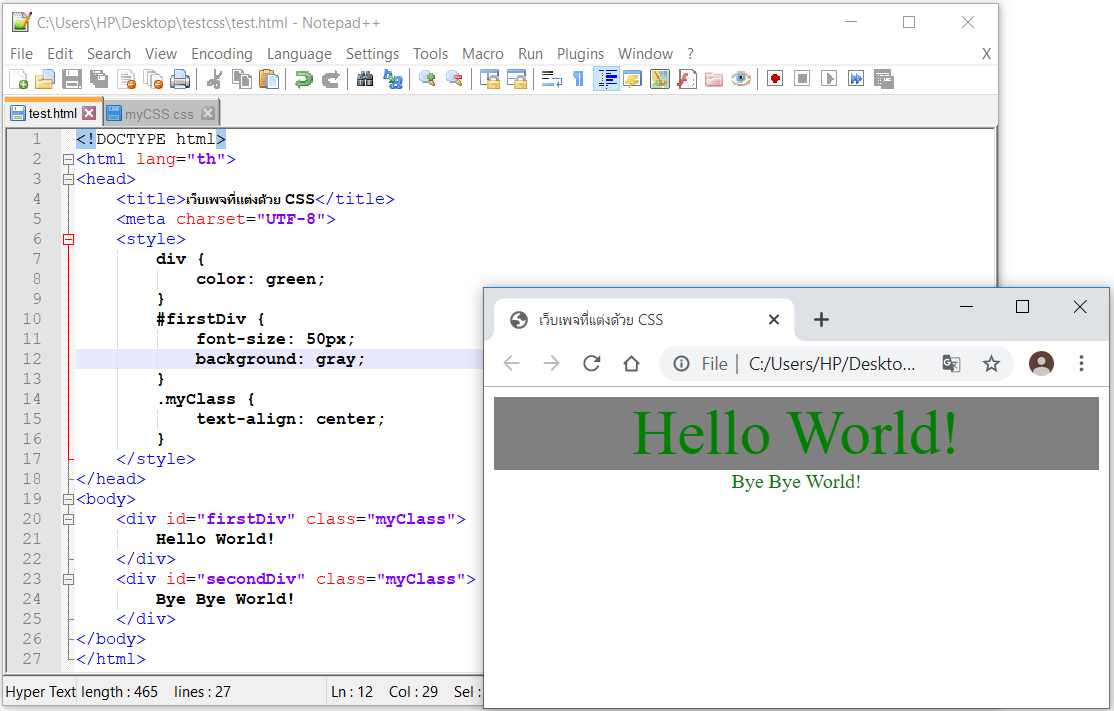
ในการจัดตำแหน่งส่วนประกอบต่างๆ บนเว็บเพจ จะอ้างอิงกับ CSS box model อยู่ตลอดเวลา ไม่ว่าจะเป็นการกำหนดระยะ margin เพื่อจัดระยะห่างระหว่างเอลิเมนต์ และการกำหนด padding เพื่อใช้กำหนดระยะห่างของข้อความภายในเอลิเมนต์นั้นๆ

**2.1.2.2 การอ้าง id และ class**

นอกจากการแก้ไข CSS ด้วยการระบุแท็กแล้ว เรายังสามารถแก้ไขโดยระบุ id หรือ class ก็ได้ โดยมีข้อกำหนดว่า หากเป็น id จะต้องนำหน้าด้วยเครื่องหมาย # ส่วนถ้าเป็น class ก็จะต้องนำหน้าด้วยเครื่องหมายจุด .

สิ่งที่ต้องทราบต่อไปนี้คือ แต่ละเอลิเมนต์ (element) บนเว็บเพจหนึ่งๆ จะมีค่า id ที่ไม่ซ้ำกันคล้ายๆ กับการที่เราทุกคน มีรหัสประจำตัวประชาชนไม่ซ้ำกันนั่นเอง แต่เนื่องจากในเว็บเพจหนึ่งๆ มักจะประกอบไปด้วยแท็กจำนวนมาก การมาเขียนโค้ด CSS เพื่อแก้ไขทีละแท็กค่อนข้างเสียเวลาวิธีที่ดีกว่า คือการใช้ CSS เข้าไปแก้ที่ class

ด้วยการที่แต่ละเอลิเมนต์ในเว็บเพจ สามารถกำหนด class ที่ซ้ำกันได้ ด้วยเหตุนี้ หากเราต้องการแก้ไขหลายๆ เอลิเมนต์พร้อมกัน จึงสามารถเข้าไปแก้ไขที่ class ดังตัวอย่างต่อไปนี้



**รูปที่** **2.5 การกำหนด id และ class**

**2.1.2.3 การเว้นระยะเอลิเมนต์**

ในเบื้องต้น เราจะพบว่ากรอบข้อความจะวางอยู่ในตำแหน่งมุมซ้ายบนของหน้าต่างเสมอ ซึ่งหากเราต้องการปรับระยะให้เลื่อนออกมาด้านขวา หรือลงมาด้านล่าง ก็ให้ใช้ในการกำหนดระยะ margin

ความแตกต่างระหว่าง padding กับ margin อยู่ตรงที่ padding เป็นระยะห่างของข้อความที่อยู่ภายในเอลิเมนต์ ส่วน margin เป็นระยะห่างระหว่างเอลิเมนต์ที่อยาภายนอก สิ่งสำคัญคือ การกำหนด margin จะไม่มีผลต่อพื้นที่ของเอลิเมนต์ที่ต้องการแสดงบนหน้าจอแต่อย่างใด

รูปแบบการเขียนโค้ดเพื่อกำหนด margin มีรูปแบบเหมือน padding ดังนี้

#mainData p {

margin: 10px 20px 15px 5px

}

สำหรับการกำหนด margin เพียงด้านเดียว เช่น กำหนดให้ห่างจากเอลิเมนต์ด้านซ้ายมือและด้านขวามือก็ให้กำหนดเป็น margin-left และ margin-right ตามลำดับ ส่วนด้านบนและด้านล่างก็ให้กำหนดเป็น margin-top และ margin-bottom

#mainData {

Magin-top: 10px

Magin-left: 10px

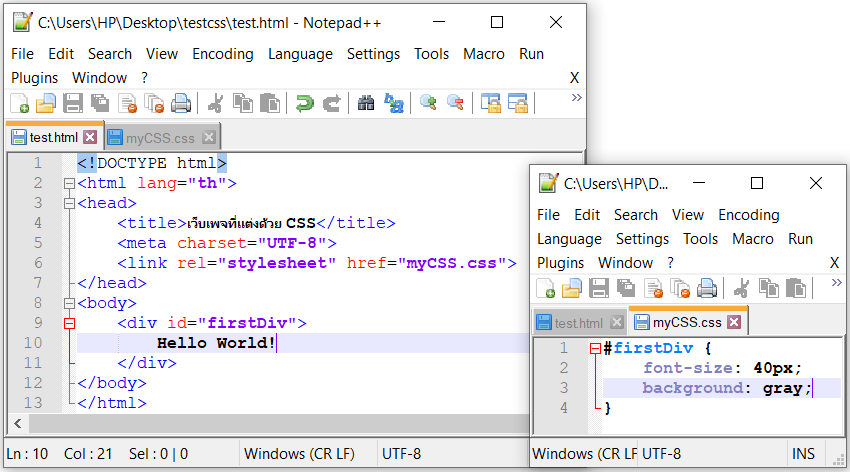
Magin-bottob: 10px

Magin-right: 10px

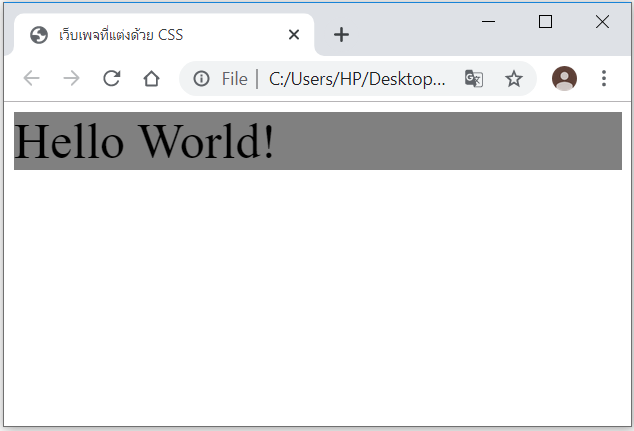
}

**2.1.2.4 การเชื่อมไปยังไฟล์ CSS**

หากต้องการเขียนโค้ดเพื่อใช้งานจริง เรานิยมสร้างไฟล์เพื่อเก็บคำสั่ง CSS โดยเฉพาะ ในภายหลังหากเว็บเพจใดต้องการใช้รูปแบบ CSS ที่ได้สร้างไว้ ก็สามารถลิงค์ไปยังไฟล์ดังกล่าวได้ทันที



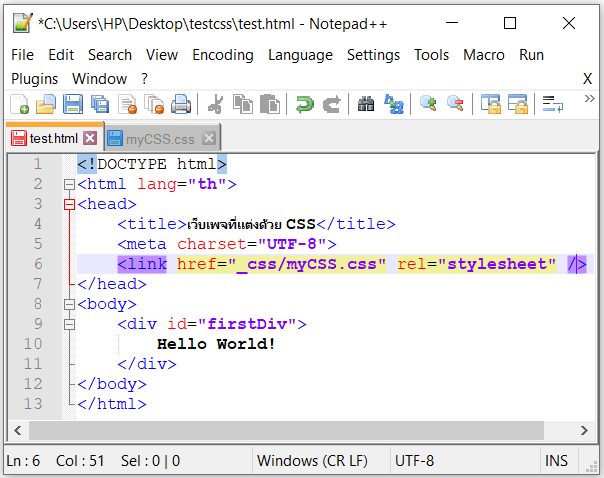
**รูปที่** **2.6 การเชื่อมไปยังไฟล์ CSS**



**รูปที่** **2.7 เบราเซอร์แสดงผลลัพธ์ของการเชื่อมไปยังไฟล์ CSS**

ในการสร้างเว็บไซต์หนึ่งๆ อาจมีเว็บเพจเป็นจำนวนมาก และแน่นอนว่าอาจมีไฟล์ที่เกี่ยวข้องมากตามไปด้วย ดังนั้น เพื่อความเป็นระเบียบและง่ายต่อการจัดการ เราจึงควรแยกเก็บไฟล์ตามประเภทของของไฟล์ลงในโฟลเดอร์ เช่น ไฟล์ CSS ก็ควรอยู่ในโฟลเดอร์ CSS ส่วนไฟล์ JavaScript ก็ควรอยู่ในโฟลเดอร์ Script แยกต่างหาก

ในกรณีที่เราเก็บไฟล์ CSS ไว้ในโฟลเดอร์อื่นๆ ที่ไม่ใช่โฟลเดอร์เดียวกับเว็บเพจ ในการลิงก์ไฟล์เราจะต้องกำหนดตำแหน่งการจัดเก็บไฟล์ให้ถูกต้องด้วย จากตัวอย่างต่อไปกำหนดลิงก์ไปยัง “\_css/myClass.css” ซึ่งหมายความว่า ไฟล์ myCSS.css ของเราเก็บอยู่ในโฟลเดอร์ \_css นั่นเอง

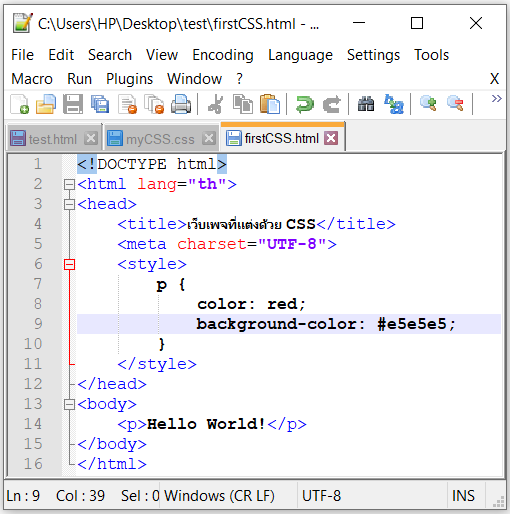


**รูปที่** **2.8 การลิงค์ไปยังโฟลเดอร์ที่จัดเก็บไฟล์**

**2.1.2.5 เขียนโค้ด CSS ลงบนเว็บเพจโดยตรง**

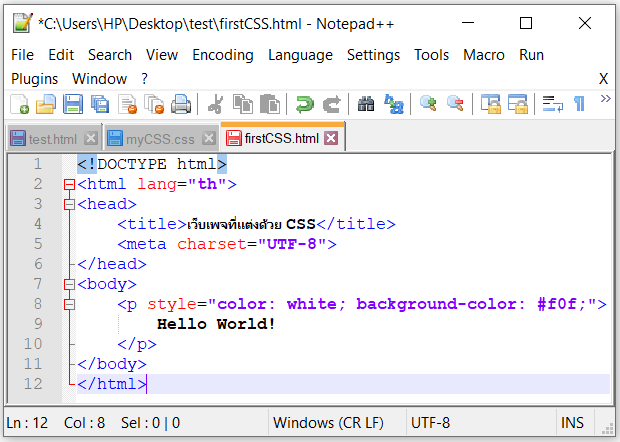
นอกจากวิธีลิงก์ไปยังไฟล์ CSS แล้ว เรายังสามารถใช้งาน CSS โดยการเขียนโค้ดลงไปในเว็บเพจได้โดยตรง ซึ่งวิธีนี้มีข้อดีคือ สะดวก แต่ก็มีข้อเสียคือ หากเราจำเป็นต้องใช้งาน CSS ใน หลายเว็บเพจ เราจะต้องเขียนโค้ดเดิมซ้ำๆ กันหลายครั้ง และเมื่อต้องการเปลี่ยนแปลงแก้ไข CSS เราจะต้องมาตามแก้ไขในทุกๆ ไฟล์ที่เกี่ยวข้อง

ในแบบแรก จะใช้แท็ก <style> </style> เพื่อครอบโค้ด CSS ดังตัวอย่างต่อไปนี้



**รูปที่** **2.9 การเขียนโค้ด CSS ลงเพจโดยตรง**

นอกจากนั้นแล้ว ในแต่ละแท็ก เราสามารถกำหนด CSS ลงไปได้ โดยใช้แอตทริ-บิว style ดังตัวอย่างต่อไปนี้



**รูปที่** **2.10 การกำหนด CSS โดยใช้แอตทริบิว style**

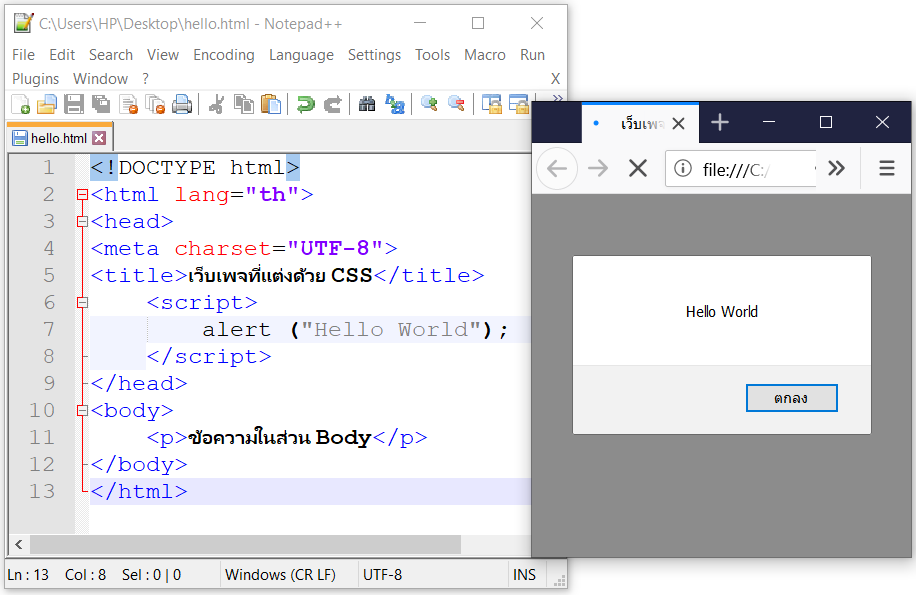
**2.1.3 ภาษาจาวาสคริปต์ (JavaScript)**

JavaScript เป็นภาษาสคริปต์ สิ่งแรกที่เราต้องทราบเกี่ยวกับ JavaScript คือ เป็นภาษาคอมพิวเตอร์ที่เขียนลงบนเว็บเพจโดยเว็บเบราว์เซอร์ (Web browser) จะเป็นผู้อ่านโค้ดและตีความหมาย แล้วจึงแสดงผลออกบนหน้าจอ จะเห็นได้ว่า JavaScript จะต้องอาศัยเบราว์เซอร์จึงไม่สามารถเข้าใช้งานฮาร์ดแวร์ต่างๆ เช่น USB, CD-ROM เป็นต้น

JavaScript เป็นสคริปต์ฝั่งไคลเอนต์ หมายความว่ามีการแปลโค้ดในฝั่งของไคลเอนต์ (Client) หรือฝั่งผู้ใช้งาน ซึ่งหากเราเขียนโค้ด JavaScript เราก็สามารถใช้เครื่องคอมพิวเตอร์ของเราทดสอบ และแสดงผลลัพธ์ได้

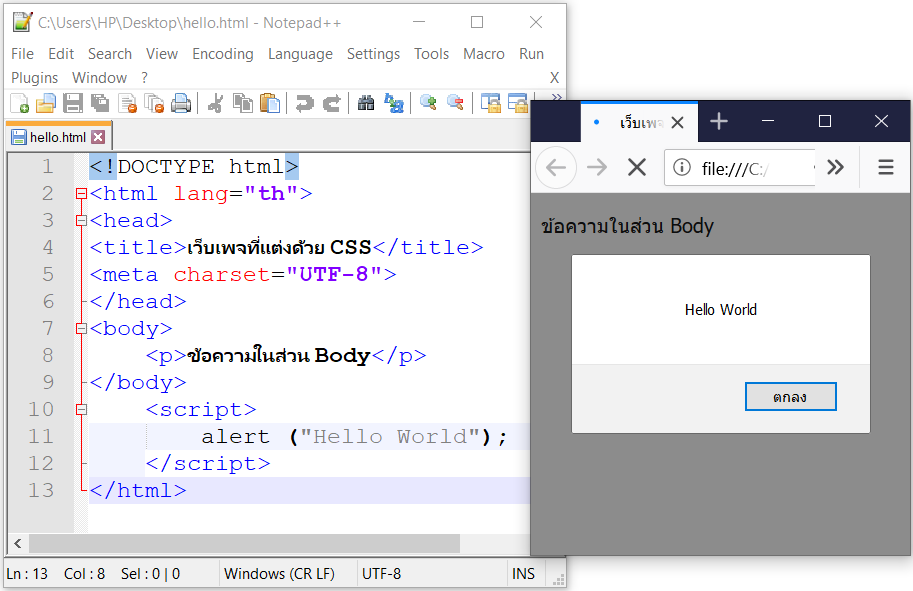
**2.1.3.1 ตำแหน่งการใส่ JavaScript**

ตำแหน่งของการใส่โค้ดคำสั่ง มีผลต่อการทำงาน เพราะเว็บเบราว์เซอร์จะแปลคำสั่งจากบรรทัดบนไล่ลงมาทีละบรรทัด ซึ่งการวางโค้ดในบรรทัดที่ต่างกัน ผลลัพธ์บนเว็บเพจก็อาจแตกต่างกันได้ดังตัวอย่างต่อไปนี้ ถ้าเราวางโค้ดในการแสดงหน้าต่างในส่วนของแท็ก <head> ข้อความภายในแท็ก <body> ก็จะไม่ถูกแสดง แต่ถ้าวางไว้ต่อท้ายจากแท็ก <body> ข้อความใน <body> จึงจะถูกแสดง



**รูปที่ 2.11** **การใส่ JavaScript ที่แท็ก <head> </head>**

จากรูปจะปรากฏหน้าต่าง Hello World แต่สังเกตที่ด้านหลัง จะไม่พบข้อความใดๆ ทั้งนี้เนื่องจากรันคำสั่ง JavaScript ก่อนที่จะแสดงข้อความในแท็ก <body>



**รูปที่** **2.12 การใส่ JavaScript ที่แท็ก <body> </body>**

ทดลองเปรียบเทียบกับการใส่ JavaScript ในแท็ก <head> และการใส่ JavaScript หลังจากแท็ก <body> จะปรากฏหน้าต่าง Hello World เหมือนกัน และถ้าสังเกตจะพบข้อความบนเว็บเพจด้วย ทั้งนี้เนื่องจากได้รันคำสั่ง JavaScript หลังจากที่แสดงข้อความในแท็ก <body> ไปแล้วนั่นเอง

**2.1.3.2 ประเภทของคำสั่ง**

ในการเขียนคำสั่ง JavaScript รวมทั้งโปรแกรมในภาษาต่างๆ จะต้องมีการเขียนคำสั่งเพื่อกำหนดค่า และควบคุมการทำงานของโปรแกรม เราเรียกคำสั่งที่ใช้เหล่านี้ว่า Statement ซึ่งสามารถแบ่งได้เป็น 3 แบบ ดังนี้

1. Assignment statement เป็นการกำหนดค่าให้กับตัวแปล หรือเขียนคำสั่งเพื่อรับค่าที่ป้อนจากแป้นคีย์บอร์ด เมาส์ หรือติดต่อกับอุปกรณ์ภายนอก เช่น การสั่งพิมพ์ หรือแสดงผลการทำงานบนหน้าจอ

2. Condition statement เป็นชุดคำสั่งที่ใช้เพื่อตรวจสอบค่า ว่าเงื่อนไขที่ตรวจสอบนั้นเป็นจริงหรือเท็จ ถ้าเป็นจริงก็ให้ทำทำงานหนึ่ง แต่ถ้าไม่เป็นจริงก็ทำอีกอย่างหนึ่ง หรือใช้เลือกว่าจะให้ทำคำสั่งไดๆ บ้าง ตัวอย่างของ Condition statement ได้แก่ การใช้งาน if

3. Iteration statement เป็นชุดคำสั่งที่สั่งให้โปรแกรมทำงานซ้ำๆ กันตามเงื่อนไขที่กำหนด เช่น สั่งให้แสดงค่าในทุกๆ ช่องของแบบฟอร์มในเว็บเพจ ตัวอย่างของ Iteration statement ได้แก่ for, while, และ do-while เป็นต้น

**2.1.3.3 ฟังก์ชัน**

คำสั่งใดที่เราใช้งานบ่อย ใช้งานซ้ำๆ นิยมนำคำสั่งเหล่านั้นสร้างเป็นฟังก์ชัน หลักการก็คือหากคำสั่งชุดใดถูกใช้งานบ่อยๆ จะถูกแยกออกเป็นส่วนย่อยๆ แล้วก็ตั้งชื่อโค้ดในส่วนนั้นเอาไว้ เมื่อใดที่ต้องการใช้งานก็ทำได้ง่ายๆ โดยการเรียกชื่อฟังก์ชัน รายละเอียดคำสั่งที่อยู่ในฟังก์ชัน ก็จะถูกเรียกขึ้นมาทำงาน

ในบางภาษาเรียกว่า โมดูล (Module) หรือบางทีก็เรียกว่า โปรซีเดอร์ (Procedure) ส่วนใน JavaScript จะเรียกว่า ฟังก์ชัน (Function) การสร้างฟังก์ชันขึ้นมาใช้งาน จะเริ่มจากการใส่คำว่า Function ตามด้วยชื่อฟังก์ชัน ตามด้วยวงเล็บ () แล้วก็ตามด้วย Code block โดยในวงเล็บ () สามารถกำหนดตัวแปลที่จะส่งค่าไปยังฟังก์ชัน หรือปล่อยเว้นว่างเอาไว้ก็ได้

function myFirstFunction () {

alert (“my function”);

// other statement, loop, another function etc.

}

เมื่อฟังก์ชันถูกสร้างขึ้นมา เราสามารถเรียกใช้งานในภายหลังได้ทุกเมื่อที่ต้องการ โดยวิธีการเรียกใช้งานฟังก์ชันทำได้ง่ายๆ โดยใส่ชื่อฟังก์ชันตามด้วยวงเล็บและ ; เช่น

myFirstFunction ();

สิ่งที่ควรทราบคือ ฟังก์ชันจะไม่ทำงานจนกว่าจะถูกเรียกใช้งาน นั่นหมายความว่า หากมีการสร้างฟังก์ชันขึ้นมา แต่ถ้าสร้างไว้เฉยๆ ไม่มีการเรียกใช้ โค้ดในฟังก์ชันก้จะไม่ถูกนำมาใช้เลย

หลักการที่ต้องจำไว้คือ เราต้องสร้างฟังก์ชันไว้ก่อนหน้าที่มีการเรียกใช้งานฟังก์ชัน เช่น สร้างฟังก์ชันไว้ในส่วน <head> แล้วเรียกใช้งานในส่วนของ <body> เป็นต้น หรือถ้าอยู่ในส่วนเดียวกันฟังก์ชันก็ควรอยู่ด้านบน

Function myFirstFunction () {

//statement 01

//statement 02

//statement 03

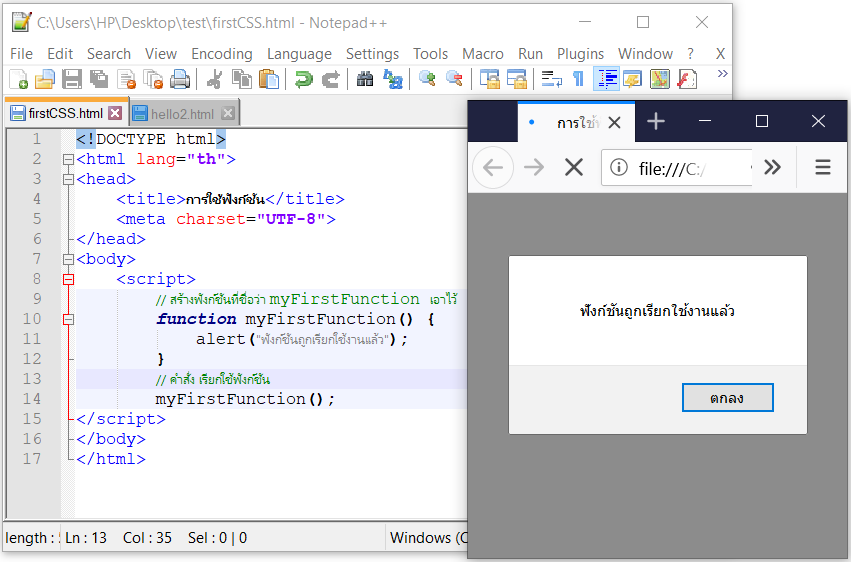
//statement 04

//…

}

myFirstFunction ();

**สรุป** ต้องกำหนดฟังก์ชันขึ้นมาก่อนที่เราจะเรียกใช้งานฟังก์ชัน และโค้ดต่อไปนี้เป็นตัวอย่างการเรียกใช้งานฟังก์ชัน

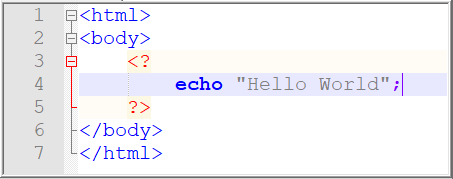


**รูปที่** **2.13 การสร้าง และเรียกใช้ function ของภาษา JavaScript**

**2.1.4 ภาษาพีเอชพี (PHP) และ มายเอสคิวแอล (MySQL)**

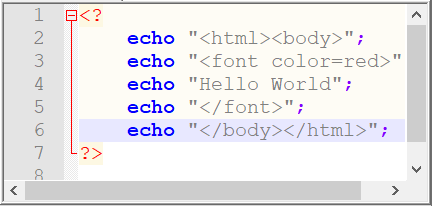
PHP เป็นโปรแกรมภาษาที่ทำงานในลักษณะ ภาษาสคริปต์ฝั่งเซิร์ฟเวอร์ (Server-side scripting language) คือมีลักษณะของการประมวลผลที่ฝั่งเซิร์ฟเวอร์ หลังจากนั้นก็จะทำการส่งผลลัพธ์ในรูปแบบ HTML กลับไปยังเว็บเบราเซอร์ ซึ่งลักษณะเช่นนี้จะทำให้ให้เราสามารถใช้ภาษา PHP พัฒนาระบบงานในลักษณะ Dynamic Programming ได้

ไฟล์ PHP จะเหมือนกับเอกสาร HTML ทั่วไป เนื่องจากเราสามารถเขียนแท็กซึ่งเป็นคำสั่งภาษา PHP ลงไปในแท็กของ HTML ได้ ไฟล์นามสกุลของ PHP คือ .php ตัวอย่างต่อไปนี้จะแสดงถึงลักษณะของการเขียนโปรแกรมภาษา PHP โดยทำการเขียนคำสั่ง PHP ให้ทำการพิมพ์ข้อความว่า Hello World และทำการแทรก PHP ไว้ในไฟล์ HTML



**รูปที่** **2.14 ภาษา PHP แทรกใน html**

ในทางกลับกันเราสามารถเขียนโปรแกรมภาษา HTML แทรกไว้ภายในภาษา PHP ได้เช่นกัน ดังตัวอย่างต่อไปนี้จะทำการพิมพ์ข้อความ Hello World เป็นตัวหนังสือสีแดงโดยใช้แท็กของ HTML เพื่อแสดงสีของตัวอักษร



**รูปที่** **2.15 ภาษา html แทรกใน PHP**

**2.1.4.1 การสร้างฟังก์ชันของ PHP**

ก่อนที่เราจะใช้งานฟังก์ชันใดๆ เราจะต้องสร้างฟังก์ชันขึ้นก่อน หลังจากนั้นค่อยเรียกชื่อฟังก์ชันนั้นเมื่อต้องการใช้งาน ซึ่งการสร้างฟังก์ชันนี้เราอาจจะเรียกว่าเป็นการประกาศฟังก์ชัน โดยรูปแบบของการประกาศฟังก์ชันในภาษา PHP เป็นดังนี้

Function ชื่อฟังก์ชัน([argument1, argyment2, …])

{

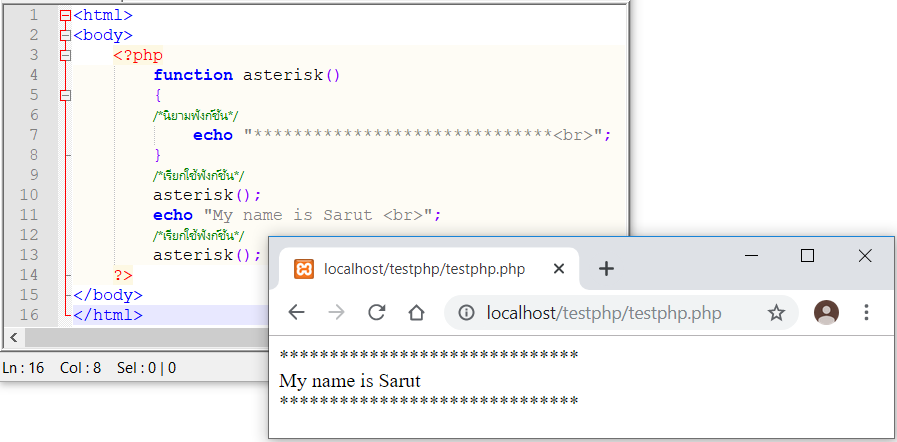
Statement;

}

**ตารางที่ 2.1 รายละเอียดของการประกาศฟังก์ชัน PHP**

|  |  |
| --- | --- |
| ชื่อฟังก์ชัน | เป็นชื่อที่ตั้งขึ้นเพื่อเรียกใช้งานฟังก์ชัน |
| argument1, argument2, … | เป็นตัวแปรที่ตั้งขึ้นมา เพื่อรอรับค่าจากการที่ฟังก์ชันส่งผ่านค่าเข้ามาประมวลผลภายในฟังก์ชัน โดยจะมีกี่อาร์กิวเมนต์ก้ได้แต่ถ้าในการใช้งานฟังก์ชันไม่มีการส่งผ่านค่าในส่วนนี้ก็ไม่ต้องระบุอาร์กิวเมนต์ เพียงวงเล็บเปิด-ปิดไว้เท่านั้น |
| Statement | เป็นคำสั่งที่ประมวลผลในฟังก์ชัน |

ยกตัวอย่างการประกาศฟังก์ชันชื่อว่า asterisk โดยเป็นฟังก์ชันที่ใช้พิมพ์เครื่องหมายดอกจัน จำนวน 30 ครั้ง รายละเอียดของฟังก์ชันนี้เป็นดังนี้



**รูปที่** **2.16 การสร้าง และเรียกใช้ function ของภาษา php**

**2.1.4.2 ฟังก์ชันของ PHP**

เราได้รู้การสร้างฟังก์ชันไว้ใช้งาน ซึ่งผู้พัฒนาจะต้องเขียนโปรแกรมขึ้นเองทั้งหมด และต้องอาศัยความรู้ในการประกาศฟังก์ชัน การเรียกฟังก์ชันใช้งาน แต่เพื่อความสะดวกโปรแกรม PHP ได้จัดเตรียมฟังก์ชันสำเร็จรูปต่างๆ ไว้ใช้งานได้โดยที่เราไม่ต้องเขียนโปรแกรมเลย เพียงแต่รู้จักชื่อฟังก์ชัน และการเรียกใช้งานฟังก์ชันให้ถูกต้อง ฟังก์ชันในลักษณะนี้มีชื่อว่า Pre-defined- Function

**ตารางที่ 2.2 ฟังก์ชันเกี่ยวกับสตริง PHP**

|  |  |
| --- | --- |
| ชื่อฟังก์ชัน | รายละเอียด |
| addslashes() | ใช้เพิ่มเครื่องหมาย \ ไว้ข้างหน้าให้กับตัวอักษรพิเศษ |
| chr() และ ord() | ใช้จัดการรหัสแอสกี้ของตัวอักษร |
| explode() และ implode() | ใช้แบ่งสตริงออกเป็นสตริงย่อยๆ และรวมสตริง |
| htmlspecialchars() | ใช้แปลงตัวอักษรพิเศษ |
| nl2br() | ใช้แทรกแท็ก <br> หรือแท็ก <br/> |
| strcmp() | ใช้เปรียบเทียบสตริง |
| strlen() | ใช้หาความยาวสตริง |
| strrev() | ใช้กลับลำดับของตัวอักษร |
| substr() | ใช้ดึงบางส่วนของสตริงออกมาจากสตริงหลัก |
| trim(), ltrim และ rtrim() | ใช้ตัดช่องว่า แท็บ ตัวอักษรขึ้นบรรทัดใหม่ |

**ตารางที่** **2.3 ฟังก์ชันเกี่ยวกับคณิตศาสตร์ PHP**

|  |  |
| --- | --- |
| ชื่อฟังก์ชัน | รายละเอียด |
| abs() | ใช้หาค่าสมบูรณ์ของตัวเลข |
| max() min() | หากลุ่มตัวเลขที่มากที่สุด, หากลุ่มตัวเลขที่น้อยที่สุด |
| pow() | ใช้หาค่าเลขยกกำลัง |
| round() | ใช้ปัดเลขทศนิยม |
| sqrt() | ใช้หาค่าที่สองของตัวเลขใด้ๆ |

**ตารางที่** **2.4 ฟังก์ชันเกี่ยวกับวันเวลา PHP**

|  |  |
| --- | --- |
| ชื่อฟังก์ชัน | รายละเอียด |
| checkdate() | ใช้ตรวจสอบวันเดือนปี |
| date() | ใช้ดึงวันเดือนปี และเวลาปัจจุบัน |
| getdate() | ให้ผลลัพธ์เป็นวันเวลาปัจจุบัน |
| time() | จะคืนค่าวัน/เวลาปัจจุบันของเครื่อง |

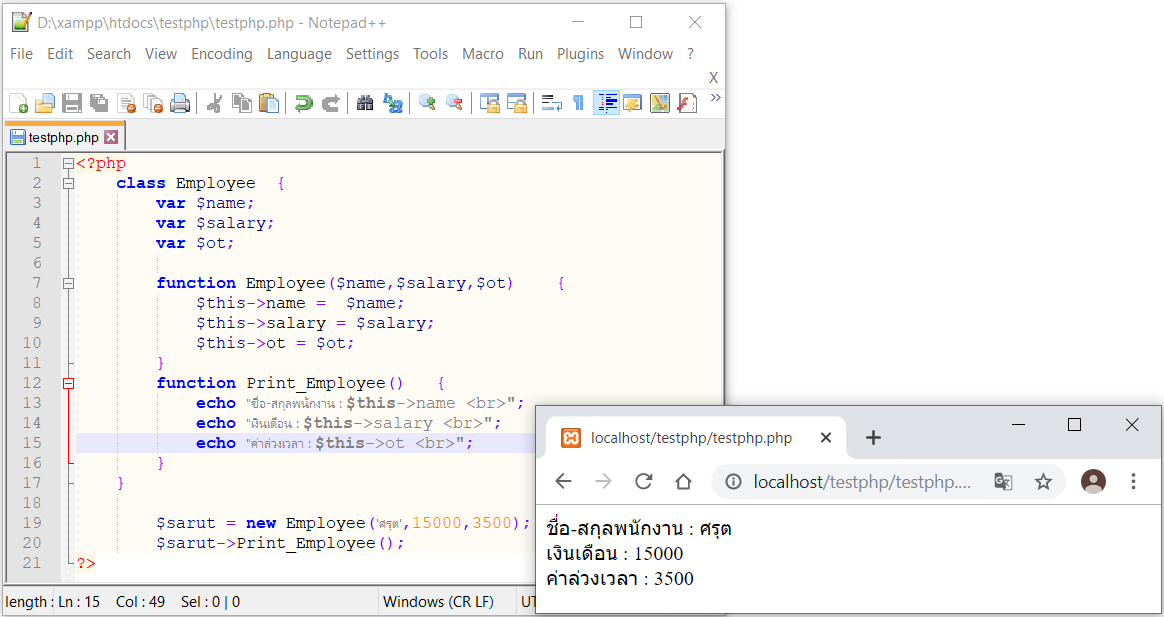
**2.1.4.3 การสร้างออบเจ็กต์และการใช้งานเมธอด**

ออบเจ็กต์ต่างๆ จะสร้างมาจากคลาส โดยเมื่อออบเจ็กต์ถูกสร้างขึ้นแล้ว ออบเจ็กต์นั้นจะมีคุณสมบัติของคลาสนั้นด้วย คือออบเจ็กต์นั้นจะมีแอตทริบิวต์ และสามารถใช้เมธอดต่างๆ ที่อยู่ในคลาสนั้นได้ รูปแบบการสร้างออบเจ็กต์ การเรียกใช้แอตทริบิวต์และการใช้งานเมธอดของออบเจ็กต์จะเป็นดังนี้

การสร้างออบเจ็กต์ $[ชื่อออปเจ็กต์ หรือตัวแปร Instance] = new ชื่อคลาส

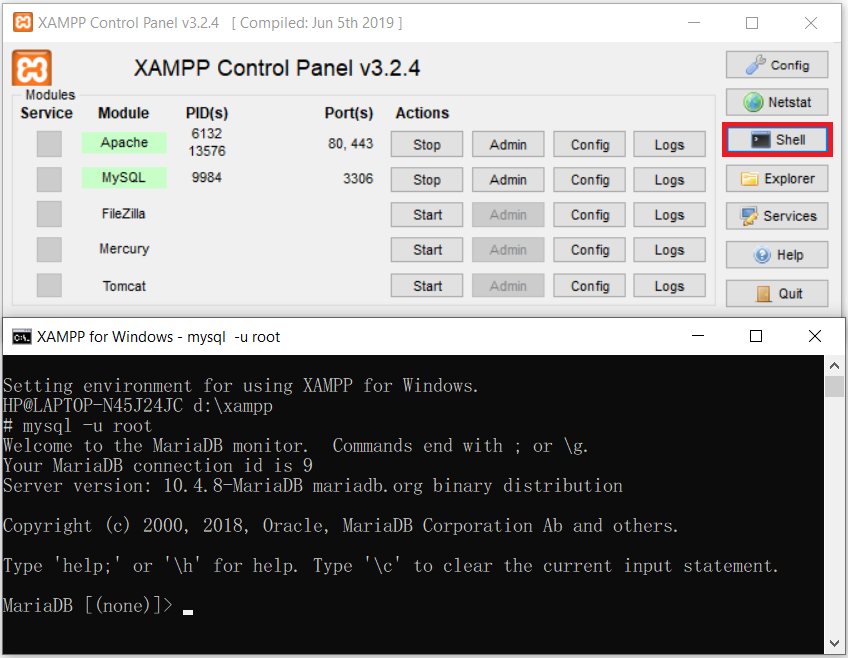
การเรียกใช้แอตทริบิวต์ $[ชื่อออปเจ็กต์ หรือตัวแปร Instance] ->Attribute

การใช้งานเมธอดต์ $[ชื่อออปเจ็กต์ หรือตัวแปร Instance] ->Method()

**รูปที่** **2.17 การสร้างออปเจ็กต์ และการใช้งานเมธอด**

**2.1.5 ฐานข้อมูล MySQL**

ฐานข้อมูลเป็นองค์ประกอบหนึ่งที่สำคัญในการพัฒนาแอพพลิเคชัน (Application) เนื่องจากระบบงานต่างๆ จะมีการจัดเก็บข้อมูลไว้ในฐานข้อมูล เพื่อให้สามารถเข้าถึงข้อมูลที่ฐานข้อมูลได้ คำว่า “เข้าถึงข้อมูล” ในที่นี้คือ การที่ระบบสามารถเพิ่ม, ลบ, แก้ไขข้อมูล และจัดทำรายงานได้ ซึ่งถือว่าเป็นฟังก์ชันพื้นฐานของระบบต่างๆ เช่น ระบบงานทางธุรกิจ เราจัดเก็บข้อมูลของสินค้าคงคลังเพื่อเก็บรายละเอียดสินค้าและสต๊อกสินค้าของบริษัท ถ้ามีลูกค้ามาสั่งซื้อสินค้า เราก็สามารถทำการตรวจสอบว่ามีสินค้าที่ลุกค้าต้องการหรือไม่ หลังจากนั้นเมื่อลูกค้าซื้อสินค้าแล้ว เราก็สามารถทำการแก้ไขหรือปรับปรุงจำนวนสินค้าที่ขายไปได้ เป็นต้น



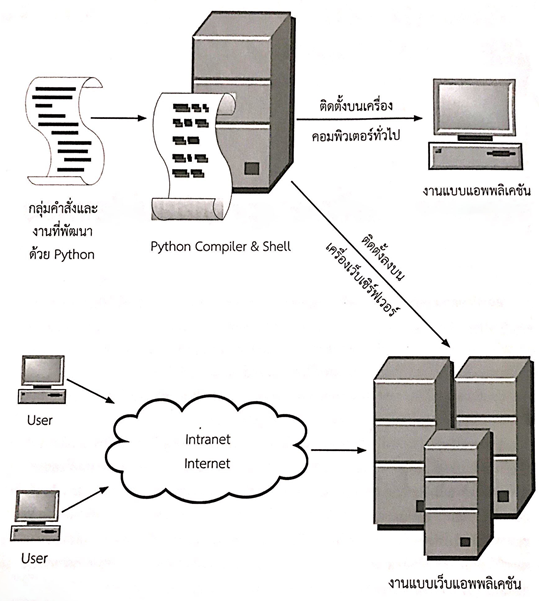
**รูปที่** **2.18 การเรียกใช้ฐานข้อมูล MySQL ใน XAMPP**

**ตารางที่** **2.5 คำสั่งเพื่อดำเนินการกับฐานข้อมูล**

|  |  |
| --- | --- |
| คำสั่ง | คำอธิบาย |
| Show database; | แสดงรายชื่อฐานข้อมูลทั้งหมด |
| Use ชื่อฐานข้อมูล; | เลือกฐานข้อมูลที่จะทำงานด้วย |
| Show table; | แสดงรายชื่อตารางทั้งหมดภายในฐานข้อมูล |
| Show colum from ชื่อตาราง;  คำสั่ง | แสดงชื่อฟิลด์และรายละเอียดของตาราง  คำอธิบาย |
| Create Database ชื่อฐานข้อมูล; | สร้างฐานข้อมูลใหม่ |
| Create Table (ชื่อฟิลด์ และรายละเอียดต่างๆ); | สร้างตารางข้อมูลใหม่ |
| Insert into ชื่อตาราง (ชื่อฟิลด์) values (ค่าของฟิลด์); | เพิ่มข้อมูลเรคอร์ดใหม่ลงในตาราง |
| Select ชื่อฟิลด์ From ชื่อตาราง [Where เงื่อนไข]; | ดึงข้อมูลจากตารางที่กำหนดและตามเงื่อนไข |
| Update ชื่อตาราง Set ชื่อฟิลด์ = ค่าที่แก้ไข [Where เงื่อนไข]; | แก้ไขข้อมูลในตารางตามเงื่อนไขที่ต้องการ |
| Delete Form ชื่อตาราง Where เงื่อนไข; | ลบเรคอร์ดจากตารางที่ระบุ และตามเงื่อนไข |
| exit | ออกจากการทำงาน MySQL |

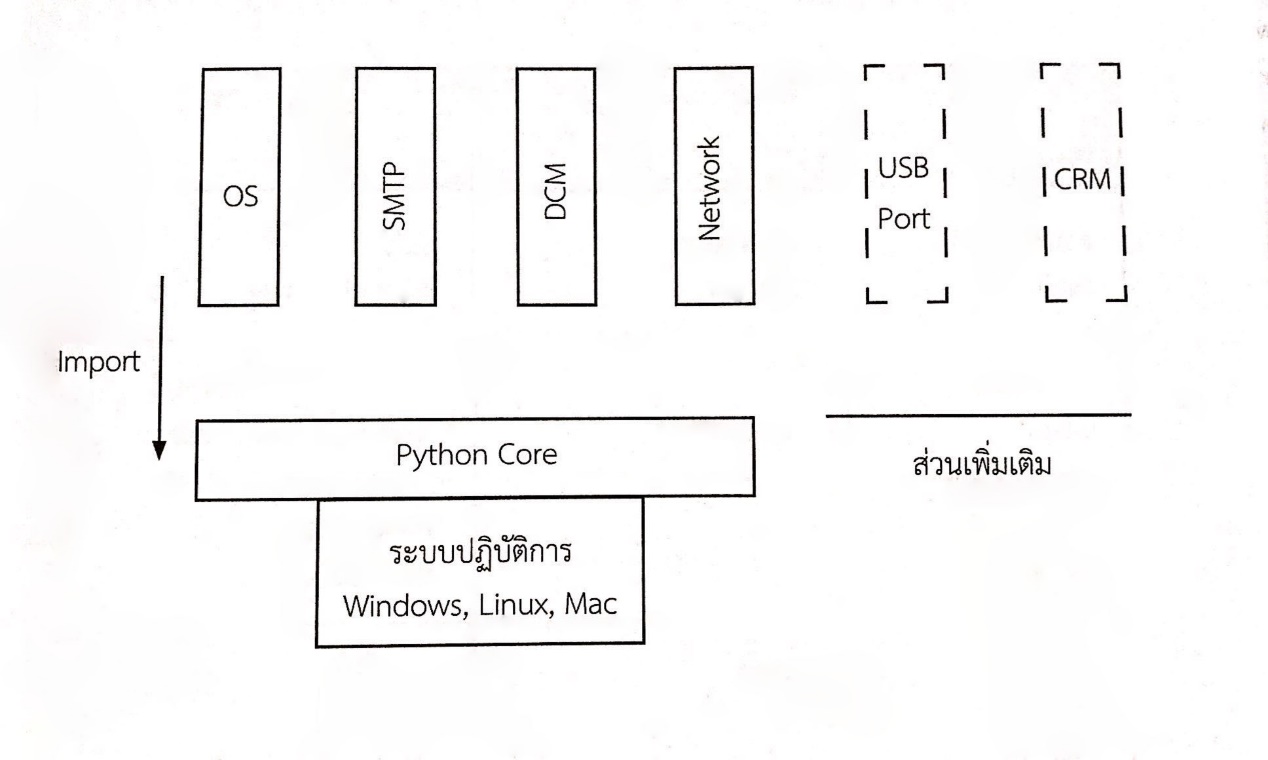
**2.1.6 ภาษาไพทอน (Python)**

สถาปัตยกรรมของ Python ได้รับการออกแบบให้นำไปพัฒนารองรับงาน แบ่งได้เป็น 2 ประเภทได้แก่ การพัฒนางานประเภทแอพพลิเคชัน (Application) และเว็บแอพพลิเคชัน (Web Application) ซึ่งวิธีการเขียนโปรแกรม การใช้คำสั่ง รูปแบบการใช้งาน เทคนิคการเขียนทั้งหลายไม่ได้แตกต่างกันมาก โดยหากนำภาษา Python ไปสร้างงานเว็บแอพพลิเคชัน ก็ต้องเพิ่มเติมความรุ้เกี่ยวกับภาษา HTML, CSS, JavaScript และการจัดการดูแลโปรแกรมบนเว็บเซิร์ฟเวอร์ จึงทำให้การเขียนเว็บโปรแกรมเป็นไปได้อย่างเหมาะสมต่อการนำไปใช้งานในแต่ละโปรเจ็กต์

****

**รูปที่** **2.19 แสดงสถาปัตยกรรมความสามารถของภาษา Python**

ขอบเขตและขีดความสามารถของ Python ในการนำไปเขียนโปรแกรมใช้งานเชิงลึกร่วมกับไลบรารีมาตราฐาน หรือบางครั้งก็เรียกว่า Standard Module ของ Python (Standard Module เป็นที่อยู่ของกลุ่มคำสั่งที่มีใช้งานใน Python โดยแบ่งตามชนิดของกลุ่มที่จะนำมาใช้งาน เช่น Standard Module เกี่ยวกับ OS ก็จะเป็นกลุ่มคำสั่งในการจัดการเกี่ยวกับ การคัดลอก (Copy), การตัด (Cut) การสร้างและคำสั่งอื่นๆ สำหรับไฟล์, โฟลเดอร์ในฮาร์ดไดรว์ของเครื่องคอมพิวเตอร์เรา เป็นต้น) ก็ชึ้นอยู่กับงานที่พัฒนาขึ้นมาต้องการให้กลุ่มคำสั่งอะไรบ้างสำหรับงานในโปรแกรมที่กำลังพัฒนา ยกตัวอย่าง เราต้องการอ่านข้อมูลจากไฟล์ไดรว์ C แล้วทำการจัดส่งอีเมล (E-mail) ให้กับลูกค้า ซึ่งในโปรแกรมของ Python จำเป็นต้องเพิ่มโมดูลเกี่ยวกับการจัดการ OS และโมดูลเกี่ยวกับอีเมล (SMTP) เพื่อเรียกใช้คำสั่งให้ครบถ้วนตามกระบวนการทำงานดังที่กำหนดไว้ในตอนต้น



**รูปที่** **2.20 โมดูลหลักที่เป็นมาตรฐานและโมดูลที่ได้เพิ่มเติมให้กับ Python**

นอกจากนี้ Python ยังเปิดโอกาสให้เพิ่มเติมโมดูลพิเศษที่นอกเหนือจากที่ Python ได้จัดเตรียมไว้ให้ ซึ่งโมดูลพิเศษเหล่านี้ก็ได้รับการพัฒนาจาก Python, C, C++, Java, Dot Net หรือภาษาอื่นๆ ด้วยเหตุผลบางประการที่จำเป็นต้องใช้ภาษาอื่นๆ มาช่วยเสริม ร่วมกับ Python ยกตัวอย่าง ถ้าต้องการต่อ Python ร่วมกับ MySQL ซึ่งคำสั่งเชื่อมโยงฐานข้อมูลพิเศษสำหรับ MySQL ไม่ได้ถูกสร้างไว้ให้เรียกใช้ในโมดูลมาตราฐานของ Python เราก็จำเป็นต้องไปดาวน์โหลด PBM Library พิเศษเฉพาะของ MySQL ที่เว็บไซต์ MySQL (หากฐานข้อมูลสนับสนุนการเชื่อมโยงกับ Python ก็จะมีไลบรารีสำหรับเชื่อมโยงฐานข้อมูลให้ดาวน์โหลดมาติดตั้งและใช้งานเพิ่มเติม)

**ตารางที่** **2.6 แสดงรายชื่อโมดูลมาตรฐานของ Python**

|  |  |  |
| --- | --- | --- |
| **กลุ่ม** | **ชื่อโมดูลมาตรฐานของ Python** | **ประเภทการใช้งาน** |
| ตัวเลข (Number) | datetime  decimal | วันที่  ตัวเลขแบบปัดพิเศษ |
| การคำนวณ (Math) | math  cmath | การคำนวณทางคณิตศาสตร์  การคำนวณจำนวนเชิงซ้อน |
| อาร์กิวเมนต์ (Argument) | sys | การผ่านค่าอาร์กิวเมนต์ของแต่ละโมดูล |
| เวลา (Time) | time | กลุ่มคำสั่งเกี่ยวกับวันเวลา |
| การสุ่มค่า | random | กลุ่มคำสั่งที่สุ่มค่าตัวเลข |
| Local Path / Directory | OS | กลุ่มคำสั่งเกี่ยวกับ Path และ Directory ของระบบปฏิบัติการ |
| ข้อมูลรูปแบบ XML | xml | กลุ่มคำสั่งจัดการดูแลข้อมูลแบบ XML |
| อีเมล | smtplip, mimetypes, e-mail, imaplib | กลุ่มคำสั่งสำหรับดูแลการจัดการอีเมล |
| เครือข่ายการสื่อสาร | socket, socketserver | กลุ่มคำสั่งดูแลจัดการโปรโตคอลและการสื่อสาร |

ทั้งนี้ภาษา Python จัดเป็นภาษาที่อยู่ในระดับสูงเทียบกับภาษา Visual Basic บางกรณีที่ต้องการใช้งานเชิงลึกในระดับ Kernal ของระบบปฏิบัติการ การเรียกใช้พอร์ตฮาร์ดแวร์ต่างๆ การจัดการระดับหน่วยความจำ การเขียนโปรแกรมระบบเครือข่ายเชิงลึก ตามที่กล่าวไว้บางส่วน ภาษา Python ไม่มีความสามารถเข้าไปจัดการได้เลย ต้องอาศัยภาษาอื่นๆ ที่มีความสามารถในการเขียนโปรแกรมเชิงลึก เช่น C, C++, Java ฯลฯ สร้างโมดูลพิเศษไว้เชื่อมต่อกับ Python ที่จะเรียกใช้ในลักษณะเป็นฟังก์ชันย่อยภายในของโมดูลพิเศษเหล่านั้นอีกที

**2.2 ฮาร์ดแวร์ที่ใช้ในการพัฒนาระบบ**

**2.2.1 Raspberry Pi**

Raspberry Pi เป็นคอมพิวเตอร์ขนาดเล็ก ออกแบบและพัฒนาโดย Raspberry Pi Foundationที่รองรับระบบปฏิบัติการ Linux Raspbian ตัวบอร์ดมี Wi-Fi, Bluetooth, USB, Ethernet Port, ช่องเสียบต่อจอภาพแบบ Micro HDMI มีช่องเสียบหูฟัง และขาอินพุต เอาต์พุต ช่อง USB สำหรับเสียบแป้นพิมพ์ เมาส์ หรืออุปกรณ์อื่น ๆ เช่น กล้อง ระบบ GPS หรือ เซนเซอร์ต่าง ๆ ฯลฯ โดยคุณสมบัติของ Raspberry pi 3 Model B สามารถแสดงได้ดังตารางที่ 3.1

**รูปภาพประกอบด้วย อุปกรณ์อิเล็กทรอนิกส์, วงจร

คำอธิบายที่สร้างขึ้นโดยอัตโนมัติ**

**รูปที่** **2.21 ตำแหน่งของอุปกรณ์ต่าง ๆ บนบอร์ด Pi3**

**ตารางที่** **2.7 คุณสมบัติของ Raspberry pi 3 Model B**

|  |  |
| --- | --- |
| Size | 86mm x 56mm x 21mm |
| Processor | Broadcom BCM2837 SoC 1.2 GHz 64-bit quad-core ARM Cortex-A53 |
| Memory | 1 GB LPDDR2 |
| Storage | MicroSD slot |
| WLAN | 802.11n |
| Ethernet | 10/100 Mbit/s |
| Bluetooth | Bluetooth 4.1, Bluetooth Low Energy (BLE) |
| GPU | Broadcom VideoCore IV |
| Video | HDMI, Composite |
| Audio | via HDMI  via 3.5 mm phone jack |
| USB | USB 2.0 4 ports |
| GPIO | 40 pins |
| Power Source | 5V via MicroUSB or GPIO header |
| Power Rating | 800 mA (4.0 W) |

**รูปภาพประกอบด้วย ภาพหน้าจอ

คำอธิบายที่สร้างขึ้นโดยอัตโนมัติ**

**รูปที่** **2.22 ตำแหน่งขา GPIO 40 PIN**

**2.2.2 DHT22 Temperature and Humidity Sensor**

DHT22 เป็นโมดูลวัดอุณหภูมิ (Temperature) และความชื้นสัมพัทธ์ (Humidity) ที่มีความแม่นยำสูงในการวัด โดยใช้ชิพDHT22 ให้ Output ออกมาเป็นแบบ Digital Output โมดูลมาพร้อม PCB และสายไฟ ต่อ R4.7K เรียบร้อยต่อสายใช้งานได้ทันที โดยมีรายละเอียดดังนี้

- ขนาด 28มม. X 12มม. X 10มม.

- รองรับแรงดันไฟฟ้าตั้งแต่ 3.3V - 6V

- วัดอุณหภูมิได้ตั้งแต่ -40 ถึง 80 องศาเซลเซียส

- วัดค่าความชื้นได้ตั้งแต่ 0 - 100%

- ค่าความผิดพลาดในการวัดอุณหภูมิเพียง +-0.5 องศาเซลเซียส

- ค่าความผิดพลาดในการวัดความชื้นเพียง 2%

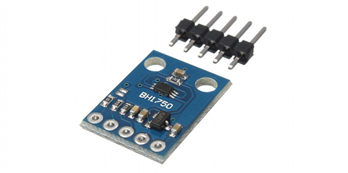
รูปภาพประกอบด้วย อุปกรณ์อิเล็กทรอนิกส์

คำอธิบายที่สร้างขึ้นโดยอัตโนมัติ

**รูปที่** **2.23 DHT22 Temperature and Humidity Sensor**

**2.2.3 Ambient Light Sensor Module GY-302 (BH1750FVI)**

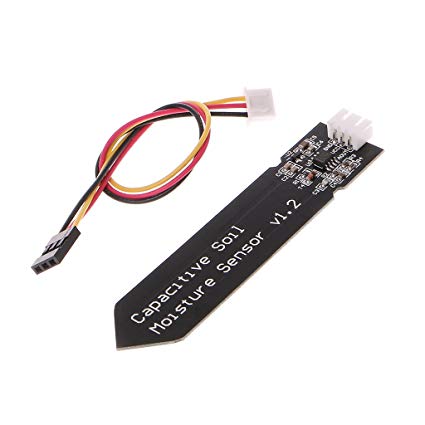
เซนเซอร์วัดความเข้มแสง BH1750FVI ให้ค่าเอาต์พุตเป็นความเข้มแสงหน่วยเป็น lux สามารถวัดค่าความเข้มแสงได้ในช่วง 0- 65535 lux ทำงานโดยใช้การสื่อสารแบบ I2C ใช้แรงดันไฟฟ้า 3 - 5V มีขนาดเพียง 13.9 มม. X 18.5 มม.



**รูปที่** **2.24 Ambient Light Sensor Module GY-302**

**2.2.4 Capacitive Analog Soil Moisture Sensor**

เซ็นเซอร์วัดความชิ้นในดินแบบ Capacitive Analog Soil Moisture Sensor นี้อาศัยหลักการแบบเดียวกับตัวเก็บประจุ คือเมื่อเซ็นเซอร์ได้รับความชื้นจะทำหน้าที่เป็นตัวเก็บประจุไฟฟ้า ยิ่งมีความชื้นมาก จะยิ่งทำให้ค่าประจุไฟฟ้ามากขึ้นเนื่องจากระยะห่างระหว่างขั้วทั้ง 2 แนบสนิทมากขึ้น โดยค่าเอาต์พุตที่ได้จะเป็นแรงดันไฟฟ้า หากใช้ Raspberry Pi จำเป็นต้องต่อ IC MCP3208 เพิ่มเพื่อให้ Raspberry Pi สามารถอ่านค่า Analog ได้ แรงดันเอาต์พุต (Vout) 0-3V แรงดันอินพุต(Vin) 3.3-5.5V



**รูปที่** **2.25 Capacitive Analog Soil Moisture Sensor**

**2.2.5 จอ LCD 16x4 Character**

เป็นจอแสดงผลขนาด 16x4 ตัวอักษร ที่แสดงผลเป็นตัวอักษรตามช่องแบบตายตัว โครงสร้างของLCDจะประกอบด้วยแผ่นแก้ว 2 แผ่นประกบกันเว้นช่องว่างตรงกลางไว้ประมาณ 6-10 โมโครเมตรด้านในจะเคลือบด้วยตัวนำไฟฟ้าแบบใส่ในระหว่างตัวนำไฟฟ้าแบบใสจะมีโมเลกุลผลึกรวมตัวกันในทิศทางที่แสงส่องผ่าน โดยจะใช้การเชื่อมต่ออนุกรม (Serial) แบบ I2Cเพื่อติดต่อสื่อสาร



**รูปที่** **2.26 จอ LCD แบบ16\*4**

**2.2.6 แผ่นร้อนเย็น (Peltier)**

แผ่นร้อนเย็น หรือ แผ่น Peltier ซึ่งเป็นอุปกรณ์ที่สามารถสร้างความเย็นได้ที่ด้านหนึ่ง และปล่อยความร้อนออกมาที่อีกด้านหนึ่ง เพียงแค่จ่ายไฟฟ้ากระแสตรงเข้าไปเท่านั้น โครงสร้างเบื้องต้นของอุปกรณ์เพลเทียร์(Peltier) ประกอบด้วยสารชนิดเอ็น (N-type)และ(P-type) ของบิสมัธ เทลลูไรด์ (สร้างมาจากสารผสมบิสมัส (Bi) กับเทลลูเรียม (Te) วางลงบนฐานรองเซรามิก ที่สร้างเป็นรูปลูกเต๋าเล็ก ๆ จำนวนมาก แล้วนำเอาแผ่นเซรามิก มาประกบไว้ทั้งสองด้าน



**รูปที่** **2.27 แผ่นร้อนเย็น หรือ แผ่น Peltier**

หลักการทำงานของแผ่นร้อนเย็น (Peltier) เป็นหลักการที่มีชื่อว่า เทอร์โมอิเล็กทริก (Thermoelectric) หลักการทำความเย็นแบบนี้เกิดขึ้นได้ โดยการใช้สารกึ่งตัวนำแบบ พี-เอ็น (P-N Type) ซึ่งสารกึ่งตัวนำแบบพี-เอ็น คือส่วนประกอบหลักของแผ่นทำความเย็นเพลเทียร์ โดยการทำความเย็นจะเกิดขึ้นได้ก็ต่อเมื่อ มีการจ่ายไฟฟ้ากระแสตรง(Direct Current : DC) หรือไฟดีซี ให้กับแผ่นทำความเย็นเพลเทียร์ เพราะเมื่อกระแสไฟฟ้าเดินทางผ่านวัสดุที่มีคุณสมบัติเป็นสารกึงตัวนำแล้วก็จะเกิดการทำปฏิกิริยาขึ้น สารกึ่งตัวนำ แบบพี-เอ็น ซึ่งต่างชนิดกัน เมื่อมีกระแสไฟฟ้าไหลผ่าน ก็จะมีการดูดกลืนกันของอิเล็กตรอนที่เคลื่อนจากระดับพลังงานต่ำทางด้านสารกึ่งตัวนำแบบพี ไปสู่ระดับพลังงานที่สูงกว่าทางด้านสารกึ่งตัวนำแบบเอ็น กระบวนการดังกล่าวส่งผลให้ที่ผิวด้านหนึ่งของแผ่นเพลเทียร์มีการดูดพลังงานความร้อน ซึ่งก็ได้จากความร้อนที่อยู่โดยรอบนั่นเอง เมื่อความร้อนในบริเวณรอบ ๆถูกดูดเข้ามา ก็จะทำให้ในบริเวณนั้นมีอุณหภูมิต่ำลง ซึ่งด้านนี้ก็คือด้านทำความเย็นนั่นเอง และในขณะเดียวกัน ก็จะเกิดการดูดกลืนของอิเล็กตรอนจากระดับพลังงานที่สูง ในสารกึ่งตัวนำแบบเอ็น สู่ระดับพลังงานที่ต่ำกว่า ในสารกึ่งตัวนำแบบพี ส่งผลให้เกิดการคายความร้อนออกมาที่บริเวณผิวหน้าของอีกด้านหนึ่ง แสดงดังรุปที่X.X

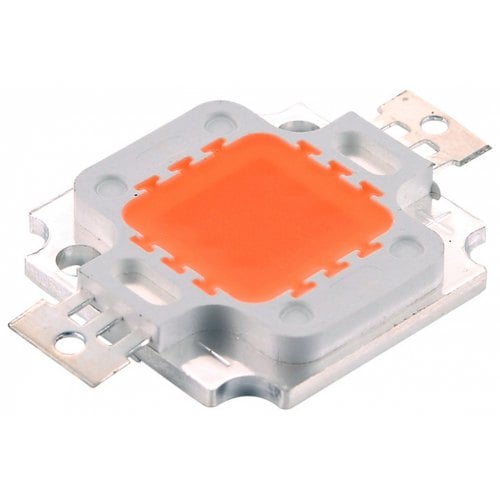
รูปภาพประกอบด้วย ข้อความ

คำอธิบายที่สร้างขึ้นโดยอัตโนมัติ  
**รูปที่** **2.28 โครงสร้างและหลักการทำงานของแผ่นร้อนเย็น (Peltier)**

จากหลักการทำงาน ทำให้สามารถนำแผ่นเพลเทียร์มาประยุกต์ใช้งานได้กับหลายๆงาน เช่น การนำคุณสมบัติในด้านการทำความเย็น เพื่อเป็นอีกหนึ่งทางเลือกสำหรับการทำความเย็นหรือลดอุณหภูมิ แบบไม่ต้องพึ่งพาระบบทำความเย็นที่ใช้คอมเพรสเซอร์ โดยทั่วไป เพลเทียร์ทำงานด้วยกระแส 4-10 แอมป์ ที่แรงดันไฟ 12 โวลต์ กระแสตรง

**2.2.7** **LED Grow Light**

หลอดไฟ LED Light Grow คือหลอดที่ผลิตขึ้นมาเพื่อใช้สำหรับปลูกต้นไม้โดยเฉพาะ ใช้สำหรับเร่งการเจริญเติมโตของต้นอ่อนป้องกันลำต้นยึด ใช้แทนแสงอาทิตย์สำหรับปลูกต้นไม้ภายในบ้าน โดยหลอดไฟ LED Light Grow นั้นจะใช้เทคโนโลยีมาใช้ในการผลิต โดยผลิต LED ขึ้นมาโดยผลิตแสงความยาวคลื่นในเฉพาะแสงย่านที่พืชมีความต้องการในการเจริญเติบโต ซึ่งคุณสมบัติพิเศษของตัวหลอด LED จะขับเฉพาะความยาวคลื่นแสงที่พืชต้องการได้ ซึ่งในกรณีนี้เราจะเลือกความยาวของคลื่นแสงที่ประมาณ 430-460nm และ 630-850nm เนื่องจากว่า ความยาวคลื่นแสงในช่วงนี้เหมาะสำหรับการสังเคราะห์แสงของต้นไม้มากที่สุด และยังช่วยในการเจริญเติบโตของต้นไม้มากที่สุด โดยจะเลือกใช้LED Grow Light ขนาด 10 W 12V ซึ่งมีรายละเอียดตามตารางที่x.x



**รูปที่** **2.29 ลักษณะของ LED Grow Light ขนาด 10 W 12V**

**ตารางที่** **2.8 รายละเอียดของ LED Grow Light ขนาด 10 W 12V**

|  |  |
| --- | --- |
| **Power** | 10W |
| **Voltage** | 9-12V DC |
| **Current** | 1000mA |
| **Color** | full spectrum |
| **Wavelength** | 400nm~850nm |
| **Luminous Flux** | 200-300LM |
| **Chip Brand** | Epistar |
| **Lifespan** | 50000 hours lifespan |
| **lighting angle** | 120~140 degree |

**2.2.8** **Switching Power Supply**

Switching Power Supply เป็นแหล่งจ่ายไฟตรงคงค่าแรงดันแบบหนึ่ง และสามารถเปลี่ยนแรงดันไฟจากไปสลับโวลต์สูง ให้เป็นแรงดันไฟตรงค่าต่ำ เพื่อใช้ในงานอิเล็กทรอนิกส์ได้ เช่น เดียวกันแหล่งจ่ายไฟเชิงเส้น (Linear Power Supply) ถึงแม้เพาเวอร์ซัพพลายทั้งสองแบบจะต้องมีการใช้หม้อแปลงในการลดทอนแรงดันสูงให้เป็นแรงดันต่ำเช่นเดียวกัน แต่สวิตชิ่งเพาเวอร์ซัพพลายจะใช้หม้อแปลงที่มีขนาดเล็ก และน้ำหนักน้อย เมื่อเทียบกับแหล่งจ่ายไฟเชิงเส้น อีกทั้งสวิตชิ่งเพาเวอร์ซัพพลายยังมีประสิทธิภาพสูงกว่าอีกด้วย



**รูปที่** **2.30 ลักษณะของ Switching Power Supply**