พื้นฐานการออกแบบและพัฒนาเว็บ

คู่มืออบรม

จัดทำโดย...

วราภรณ์ วิยานนท์

เอกสารประกอบการอบรม

สำหรับผู้ดูแลเว็บหน่วยงานของมหาวิทยาลัย

สำนักคอมพิวเตอร์ มหาวิทยาลัยศรีนครินทรวิโรฒ
กรกฎาคม 2547

สารบัญ

พื้นฐานการออกแบบและพัฒนาเว็บ

คู่มืออบรม

การออกแบบและพัฒนาเว็บที่ดีนั้นต้องอาศัยเทคโนโลยีและความรู้หลายด้านประกอบกันเพื่อให้เว็บที่สร้างนั้นสามารถเข้าถึงได้จากกลุ่มคนหลายประเภท ไม่ว่าจะเป็นบุคคลธรรมดาหรือคนที่มีความผิดปกติด้านร่างกายก็ตาม หรือแม้กระทั่งการเข้าถึงของโปรแกรมค้นหาทั่วไปที่จะเข้ามาสำรวจเนื้อหาในเว็บหน้าต่างๆเพื่อนำไปสร้างดัชนี

ความรู้พื้นฐานที่ผู้พัฒนาเว็บควรทราบ

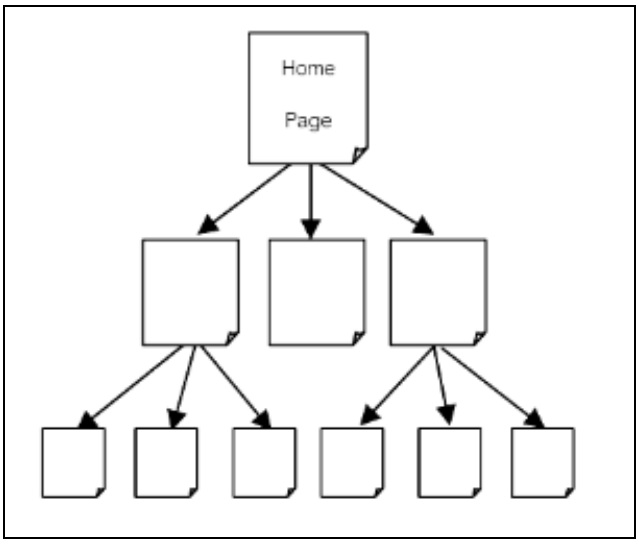
- การออกแบบเว็บเพจ
- HTML
- การใช้สไตล์
- การออกแบบเว็บโดยใช้หลักการ include
- การโปรแกรมภาษา (สำหรับการสร้างเว็บเพจแบบ dynamic)

ในเอกสารฉบับนี้อาจจะนำเสนอแต่ละหัวข้อไม่ละเอียดนัก แต่จะพยายามเน้นถึงจุดสำคัญที่ควรคำนึงถึงในการออกแบบและพัฒนา

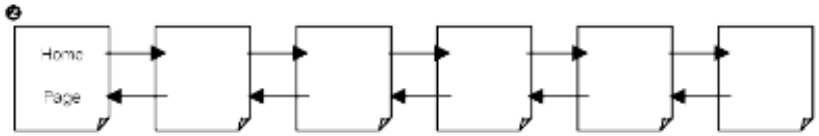
หลักการออกแบบเว็บเพจ

การออกแบบและพัฒนาเว็บเพจ สามารถทำได้หลายระบบ ขึ้นอยู่กับลักษณะของข้อมูล กลุ่มเป้าหมายที่ต้องการนำเสนอ เช่น หากกลุ่มเป้าหมายเป็นเด็กวันรุ่น และนำเสนอข้อมูลเกี่ยวกับความบันเทิง อาจจะออกแบบให้มีทิศทางการไหลของหน้าเว็บที่หลากหลายใช้ลูกเล่นได้มากกว่าเว็บที่นำเสนอให้กับผู้ใหญ่ หรือเว็บด้านวิชาการ ทั้งนี้หลักการออกแบบเว็บเพจสามารถแบ่งได้ 3 ลักษณะคือ

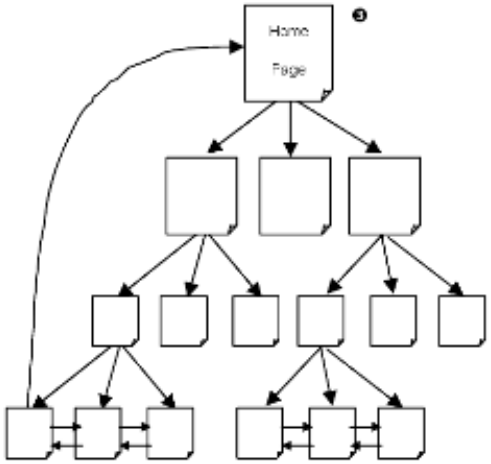
1. แบบลำดับขั้น (Hierarchy) เป็นการจัดแสดงหน้าเว็บเรียงตามลำดับกิ่งก้านแตกแขนงต่อเนื่องไปเหมือนต้นไม้กลับหัว เหมาะสำหรับการนำเสนอข้อมูลที่มีการแบ่งเป็นหมวดหมู่ที่ไม่มากนัก และมีข้อมูลย่อยไม่ลึกมาก



2. แบบเชิงเส้น (Linear) เป็นการจัดแสดงหน้าเว็บเรียงต่อเนื่องไปในทิศทางเดียว เหมาสำหรับการนำเสนอข้อมูลที่เป็นรูปภาพ เช่น Presentation ต่างๆ



3. แบบผสม (Combination) เป็นการจัดหน้าเว็บชนิดผสมระหว่างแบบลำดับขั้นและแบบเชิงเส้น มันจะเป็นที่นิยมใช้กันอย่างแพร่หลายในปัจจุบัน เนื่องจากสามารถควบคุมการนำเสนอและการเรียกดูได้สะดวกและรวดเร็ว



HTML

HTML หรือ HyperText Markup Language เป็นภาษา script ประเภทหนึ่ง ซึ่งใช้ทำ Web page เป็นงานหลักในระบบ World Wide Web ในแรกเริ่ม วัตถุประสงค์หลักของ HTML ถูกเสนอโดยนาย Berners-Lee ซึ่งเป็นนักโปรแกรมเมอร์ทำงานที่ European Center for Particle Physics (CERN) ได้กำหนดไว้ว่า

- เพื่อสร้างสื่อที่นักวิทยาศาสตร์สามารถจะเผยแพร่ผลงาน และใช้อ้างอิง ได้ตลอด 24 ช.ม.
- เพื่อสร้างภาษาคอมพิวเตอร์ที่รองรับภาษาท้องถิ่น ที่ไม่ขึ้นกับระบบของเครื่องคอมพิวเตอร์ (Platform) หรือระบบเครือข่ายใดๆ

และด้วยวัตถุประสงค์ข้างต้น ภาษา HTML จึงถูกใช้งานอย่างแพร่หลายในสังคมของนักวิทยาศาสตร์ และกำหนดให้เครื่องมือที่ใช้เขียน เป็นโปรแกรม text editor ทั่วๆ ไป

สำหรับภาษา HTML ในอินเทอร์เน็ต ได้รับการพัฒนาอย่างต่อเนื่อง เพื่อให้คนทุกๆ ชาติบนโลก สามารถเข้าถึง เผยแพร่ และอ้างอิง วิทยาการความรู้ได้ ด้วยการเชื่อมโยงไปมาแบบ hyperlink อาจจะด้วยตัวอักษร และ/หรือ รูปภาพ โดยอาจเชื่อมโยงเฉพาะภายใน เอกสารนั้น หรือเชื่อมโยงข้ามไปยังเอกสารอื่นๆ ได้

ภาษา HTML มีต้นแบบมาจากภาษา SGML (Standard Generalized Markup Language) ซึ่งเป็นภาษาที่ใช้ได้เฉพาะ กับประเภท ของคอมพิวเตอร์ และสิ่งที่ HTML รับมาจาก SGML คือ การประกาศค่า และ การกำหนดรูปแบบเอกสาร (Document Type Definition --DTD)

สิ่งที่ทำให้ ภาษา HTML ได้รับความนิยมอย่างมาก และรวดเร็ว ก็คือ HTML รวมถึง โปรโตคอล HTTP (HyperText Transfer Protocol) เป็นภาษาที่ใช้สื่อสารกันได้ทั่วโลก โดยที่ตัวภาษาและโปรโตคอล ไม่ขึ้นกับระบบเครือข่าย และประเภทของคอมพิวเตอร์ (Platform) ซึ่งมีความหลากหลาย อันเนื่องมาจากเทคโนโลยีและประเภทการใช้งาน เป็นผลให้เอกสารที่เขียนโดย HTML สามารถถ่ายโอน ได้อย่างกว้างขวาง ทั้งในรูปแบบของ ตัวอักษร ภาพ และเสียง

มาตรฐาน HTML เวอร์ชัน 4.0 ถูกกำหนดโดยคณะกรรมการชุดหนึ่ง ที่มีชื่อว่า The World Wide Web Consortium (W3C) โดยจัดแบ่งเอกสารที่เขียนด้วยภาษา HTML (เรียกว่า เอกสารแบบ Hypertext) ออกเป็น 3 ประเภทได้แก่

1. แบบเคร่งครัด (Strict HTML 4.0)

   เป็นเอกสาร Hypertext ที่เขียนด้วยภาษา HTML 4.0 ตามมาตรฐานอย่างเคร่งครัด tag ใดที่คณะกรรมการชุดนี้ นิยามว่า ล้าสมัย (Deprecate) หรือ ให้เลิกใช้ (Obsolete) ก็จะไม่ใช้คำสั่งนั้นในการเขียนเอกสาร ซึ่งในความเป็น จริงขณะนี้ ยังคงไม่มี โปรแกรม browser ใด สนับสนุนภาษา HTML 4.0 อย่างเคร่งครัด แต่คาดว่าในอนาคต อันใกล้ น่าจะมีความเป็นไปได้

2. แบบค่อยเป็นไป (Transitional/ Loose HTML 4.0)

   เป็นเอกสารที่สร้างด้วยภาษา HTML 4.0 โดยใช้ร่วมกับคำสั่งใน HTML เวอร์ชัน 3.2 เพื่อให้เอกสารที่สร้างขึ้นมีรูปแบบ และใช้งานได้ตามจริง แม้ว่าจะใช้โปรแกรม browser ระบบเครือข่าย และประเภท คอมพิวเตอร์ที่หลากหลายก็ตาม

3. แบบ Frameset (Frameset HTML 4.0)

   เป็นเอกสารที่รวมเอาประเภท Transitional เข้ากับ tag ประเภท frame (ได้แก่ FRAME, FRAMESET, NOFRAMES และ IFRAME) ซึ่งเป็น tag ใหม่เพิ่งจะมีในเวอร์ชัน 4.0 นี้

- **ซอร์ฟแวร์ช่วยเขียน HTML**

หากเราเขียนเว็บเพ็จด้วยภาษา HTML โดยตรง ซอร์ฟแวร์ที่ต้องใช้คือ โปรแกรม text editor ธรรมดาๆ ไม่ว่าเราจะใช้ระบบปฏิบัติการ แบบใด (DOS/ Windows, UNIX/ Linux หรือ MacIntosh) ก็สามารถสร้าง เว็บเพ็จหรือเอกสารแบบ hypertext ได้ อย่างไรก็ตาม สำหรับผู้ต้องการสร้างเว็บเพ็จของตนอย่างรวดเร็ว และไม่ต้องการเรียนรู้ภาษา HTML โดยตรง ก็สามารถหาซอร์ฟแวร์เพื่อทุ่นแรง ทุ่นเวลามาใช้แทนได้

เราสามารถจัดกลุ่มซอร์ฟแวร์ สำหรับช่วยสร้างเว็บเพ็จออกเป็น 2 ประเภทด้วยกัน คือ

- โปรแกรมช่วยเขียน tags ภาษา HTML
- โปรแกรมแบบ WYSIWYG

โปรแกรมทั้งสองประเภท ต่างมีข้อดี และข้อด้อยเป็นของเฉพาะตัว หากเราต้องการผลิตเว็บเพ็จที่สวยงาม มีความยืดหยุ่นสูง เป็นไป ตามความต้องการของเรามากที่สุด ก็จำเป็นต้องสร้างเว็บเพ็จขึ้นจากภาษา HTML โดยตรง

โปรแกรมช่วยเขียน Tags

ปัจจุบันนี้ มีซอร์ฟแวร์ช่วยเขียน tags ในภาษา HTML ออกมาหลากหลายโปรแกรม แต่ที่จะแนะนำต่อไปนี้ ได้รับการยอมรับจากผู้สร้าง เว็บเพ็จทั้งในระดับมือสมัครเล่น และมืออาชีพ แล้วว่าค่อนข้างดี มีความยืดหยุ่นสูง ได้แก่ Allaire's HomeSite, Allaire's ColdFusion, HotDog Pro, HTML Assistant Pro และ CoffeeCup HTML Editor++

โปรแกรมประเภทนี้ จะมีประโยชน์มาก โดยเฉพาะผู้มีความรู้ในภาษา HTML ทั้งนี้เพราะ ผู้สร้างเว็บเพ็จ ที่มีความรู้ในภาษา HTML จึงจะ ทราบว่าต้องใส่ tags อะไรบ้างจึงจะได้เว็บเพ็จ ออกมาในรูปแบบที่ต้องการ เช่น เมื่อต้องการขึ้นย่อหน้าใหม่ ด้วยตัวอักษร ขนาด 16 point Time New Roman สีน้ำตาล เราต้องใส่ tags กำหนดให้ขึ้นย่อหน้าใหม่ <p> ตามด้วย tag กำหนดลักษณะตัวอักษร <font> เป็นต้น

ดังนั้น ผู้ใช้โปรแกรมประเภทนี้ จะเป็นผู้ที่มีความรู้ในภาษา HTML เป็นส่วนใหญ่ แต่ต้องการผลิตเว็บเพ็จ ให้รวดเร็วขึ้น โดยที่ผู้เขียน ไม่ต้องพิมพ์ tags ด้วยตนเอง และสามารถหาข้อผิดพลาดที่เกิดขึ้นได้ง่าย เพราะ โปรแกรมประเภทนี้จะกำหนดสีให้ tags ตลอดจน Attributes และ Values แตกต่างไปจาก สีที่เป็นเนื้อหาของเอกสาร นอกจากนี้ โปรแกรมประเภทนี้ยังตระเตรียม script ที่ใช้กัน บ่อยๆ ไว้ให้ เราเพียงแค่ใส่ tag กำหนด script ไว้เท่านั้น

โปรแกรมประเภท WYSIWYG

โปรแกรมประเภท WYSIWYG (What You See Is What You Get) เป็นซอร์ฟแวร์ประเภทกราฟิก ทำหน้าแปล สิ่งที่กำหนด ในรูปของกราฟิก ขนาดและรูปแบบของตัวอักษร สี ย่อหน้า รูปภาพ ฯลฯ ให้เป็น tags ในภาษา HTML ผู้ใช้โปรแกรมเหล่านี้ ไม่มี ความจำเป็นต้องเรียนรู้ภาษา HTML เลย หากแต่จำเป็นต้องเรียนรู้วิธีใช้ซอร์ฟแวร์ดังกล่าว โปรแกรมประเภทนี้ ได้แก่ MS FrontPage, Adobe GoLive!, และ Macromedia's Dreamweaver เป็นต้น

• โครงสร้าง HTML

โครงสร้างของภาษา HTML ประกอบไปด้วย tag เป็นคู่ๆ จากตัวอย่าง <HTML> และ</HTML> เป็น tag ที่ประกาศว่า เอกสารต่อไปนี้จะ ถูกเขียนขึ้นด้วยภาษา HTML และภายในจะเป็น tag คำสั่งของ HTML

```
<HTML>
<HEAD>
:
</HEAD>
<BODY bgcolor="#FFFFFF">
:
</BODY>
</HTML>
```

ทุกครั้งที่จะสร้างเอกสารด้วย HTML จะต้องเขียนโครงสร้างข้างต้น โดยมีหลักการจำคือ เริ่มด้วย tag ของ HTML ตามด้วย tag ของ ส่วนหัว (<HEAD> และ </HEAD>) และตัว (<BODY> และ </BODY>) ตามลำดับ กล่าวคือ มีลักษณะคล้ายคน คือมีหัว และตัว

ส่วนหัวของโครงสร้าง HTML บรรจุข้อมูลต่อไปนี้

Page Title คือ ส่วนที่จะปรากฏอยู่บน Title bar ของหน้าต่างบนโปรแกรม Web browser

Script คือ ส่วนที่เราเขียน java script หรือ script ภาษาอื่นๆ เพื่อเพิ่มสีสรรให้กับเว็บเพ็จของเรา

Style คือ คำสั่งเพื่อให้เว็บเพ็จหน้านั้น นำเอารูปแบบ ต่างๆ ที่กำหนดใน Cascading Style Sheet มาใช้งาน

META เป็นส่วนที่ผู้เขียน จะให้ข้อมูลแก่โปรแกรม Web browser และ Search engines ต่างๆ เกี่ยวกับเว็บเพ็จหน้านั้น ซึ่งข้อมูลในส่วนนี้ แม้ว่าจะไม่ปรากฏในหน้าต่างของโปรแกรม Web browser ก็ตาม แต่ก็มีความสำคัญไม่ยิ่งหย่อนไปกว่ากัน รายละเอียดของ tag เหล่านี้จะได้กล่าวโดยละเอียดต่อไป

ส่วนตัวของโครงสร้างฯ บรรจุ เนื้อหา ข้อความ รูปภาพ หรือ แม้กระทั่งไฟล์เสียงเพลง (เพื่อ download)

จากโครงสร้างข้างต้น เราจำแนกโครงสร้างของคำสั่งออกเป็น 3 ส่วนด้วยกัน คือ

1. Tags คือ คำสั่งในภาษา HTML โดยจะมีประสิทธิภาพสูงสุด เมื่อกำหนดส่วนขยายให้กับ tags
2. Attributes เป็นส่วนประกอบหนึ่งของส่วนขยาย ทำหน้าที่กำหนดทิศทางของ tags
3. Values เป็นส่วนประกอบสุดท้ายของส่วนขยาย ทำหน้าที่กำหนด ขนาด หรือ ลักษณะ ให้กับ attributes

หากเราสังเกตโครงสร้างคำสั่งข้างต้น เราพบว่า ไม่จำเป็นที่ทุกคำสั่งในภาษา HTML จะต้องมีส่วนประกอบครบทั้ง 3 ส่วน แต่ส่วนใหญ่แล้ว จะมี เพื่อให้โปรแกรม web browser สามารถแสดงรูปแบบเอกสารได้อย่างที่เราต้องการ

- **ไม่ควรลืมในการทำงานร่วมกับ HTML Tag ต่างๆ**

TITLE

แท็ก title เป็นส่วนสำคัญสำหรับโปรแกรมค้นหาประเภท Search Engine ที่จะ title มาสร้างเป็น index และ bookmark Title ในแต่ละเพจควรแตกต่างกันในแต่ละเพจและควรมี descriptive name ในเพจแรกของเว็บไซต์ ตามด้วยชื่อหน่วยงานและมหาวิทยาลัยศรีนครินทรวิโรฒ ข้อความใน Title นั้นควรเป็นข้อความสั้นหรือมีความยาวไม่เกิน 60 ตัวอักษร และควรหลีกเลี่ยงอักขระพิเศษ เช่น เครื่องหมายคำถาม เป็นต้น

ชื่อ Title ไม่ควรเป็นข้อความต้อนรับ เช่น ขอต้อนรับเข้าสู่หน่วยงาน... เป็นต้น ควรเป็นชื่อที่เริ่มต้นด้วยคำอธิบายเนื้อหาในเว็บเพจโดยเฉพาะ และจบด้วยคำอธิบายทั่วไป ตัวอย่างเช่น

<title>หลักสูตรบัณฑิตศึกษา – บันทิตวิทยาลัย – มหาวิทยาลัยศรีนครินทรวิโรฒ **</title>**

META

การใช้แท็ก Meta Keyword และ Meta Description ช่วยให้เว็บไซต์สามารถถูกค้นพบได้ คีย์เวิร์ดและคำอธิบายควรสอดคล้องกับเนื้อหาของเว็บเพจนั้นๆ

แท็ก Meta keyword อธิบายเนื้อหาในเว็บเพจ ควรใช้เป็นวลี 2-3 วลีที่คั่นด้วยเครื่องหมายลูกน้ำ (comma) ขนาดของคำอธิบายควรอยู่ระหว่าง 300 – 500 ตัวอย่างเช่น

<meta name=**"keywords"** content="Srinakharinwirot University, Humanity Faculty, Science Faculty, Graduate School, Computer Center"**>**

ผลลัพธ์ที่แสดงรายละเอียดจากการค้นหาจะนำเนื้อหามาจาก Meta description แต่ถ้าหากเนื้อหาใน Meta description ยาวเกินไปโปรแกรมค้นหาอาจจะไม่แสดง ดังนั้นความยาวของ description ไม่ควรเกิน 200 อักษร ตัวอย่างเช่น

<meta name=**"description"** content="……."**>**

IMG

เราใช้ <IMG> เพื่อใส่รูปและกราฟิก ลงบนเว็บเพ็จ ซึ่งเราต้องใช้ tag นี้เดี่ยวๆ โดยไม่มี tag ปิด นอกจากนี้ <IMG> ยังต้องกำหนด แหล่งที่เก็บของไฟล์กราฟิก ด้วย SRC ดังนี้

<IMG SRC="image.gif"**>**

Image Attributes

นอกจาก เราต้องกำหนดแหล่งที่มาของรูปภาพ แล้ว เรายังต้องกำหนด attributes ต่างๆ ให้กับ image tag เพื่อให้ image tag ทำงานเต็มประสิทธิภาพ

1. width="x" ใช้กำหนดความกว้างของรูปหรือกราฟิก ในหน่วยของ pixels
2. height="x" ใช้ร่วมกับ width เพื่อกำหนดความสูงของรูป ในหน่วยของ pixels
3. border="x" ใช้กำหนดกรอบของรูปหรือกราฟิกให้มี ความหนา-บาง ในหน่วยของ pixels เช่นกัน
4. align="x" ใช้กำหนดตำแหน่ง (alignment) ของรูป
5. alt="description of image" ใช้อธิบายความหมายของรูป เพื่อ ให้ผู้ใช้ web browser แบบ text ได้ทราบว่าสิ่งที่เขามองไม่เห็น เป็นรูปของอะไร
6. hspace="x" ใช้กำหนดพื้นที่ว่าง (white spaces) ในแนวราบ ระหว่างกราฟิกกับส่วนอื่นๆ (อาจเป็นข้อความ หรือ รูปภาพที่ อยู่ข้างๆ กัน)
7. vspace="x" ใช้กำหนดพื้นที่ว่างในแนวดิ่ง ระหว่างกราฟิกกับ ส่วนอื่นๆ

คำแนะนำ

1. Width และ Height

   ควรที่เราจะกำหนด attributes ทั้งสองไว้ใน <IMG> เสมอ ทั้งนี้ attributes ทั้งสองจะช่วยให้ browser จัดการกับรูปหรือกราฟิกได้ อย่างถูกต้องตลอดทั้งหน้าเว็บเพ็จ และไม่ควรใช้ width และ height กำหนดขนาดกราฟิกที่ผิดไปจาก ความจริง ค่าที่กำหนดต้องเป็น ขนาดจริงของรูปเสมอ

2. Alternative

   Alt attribute ทำหน้าที่อธิบายความหมายของ รูปกราฟิกให้กับผู้ใช้ web browser แบบ text ได้ทราบ โดยที่ alt attribute จะแสดง ข้อความตามที่เขียนอธิบายไว้ เราเพียงเติม alt attribute ต่อท้าย attribute ก่อนๆ ดังนี้

<IMG SRC="images/book.gif" width="63" height="70" border="1" alt="หนังสือ">

ในขณะรอโหลดรูปกราฟิก Browser จะ แสดงตำแหน่งและขอบเขตของกราฟิกพร้อมด้วยข้อความที่เราเขียนอธิบายไว้ภายในกรอบรูป นอกจากนี้ เมื่อโหลดรูปเสร็จ ลองเอาเมาส์ไปจิ้มดู ก็จะเห็นคำอธิบายไว้ปรากฏขึ้นเช่นกัน

encoding

ปัญหา encoding คือ ปัญหาที่เห็นข้อมูลบนเว็บเป็นภาษาต่างด้าว ลักษณะเป็น &#nnnn; เรียงต่อกันแทนคำที่เราพิมพ์!ทำให้อ่าน(ภาษาไทย)ไม่ออก
สาเหตุเกิดจาก การบังคับให้บราวเซอร์แสดงเป็นภาษาที่ผู้เข้าชมต้องการให้โดยอัตโนมัติ(เช่น ภาษาไทย) โดยกำหนดให้เป็นตัวอักษรหลัก(charset- character set) ได้แก่ iso-8859-1 สำหรับภาษา Western European languages เช่น ภาษาอังกฤษ(English), tis-620 สำหรับภาษาไทย เป็นต้น แต่บราวเซอร์ไม่เปลี่ยน charset ไปตามภาษาที่ต้องการอย่างอัตโนมัติ

<meta http-equiv="Content-Type" content="text/html; **charset=TIS-620">**

หรือ

<meta http-equiv="Content-Type" content="text/html; **charset=windows-874">**

การใช้สไตล์ (Cascading Style Sheets)

Cascading Style Sheet (CSS) เป็นคำกว้างๆ ที่ใช้อ้างถึงวิธีการกำหนดรูปแบบ (Format) ของเอกสาร HTML ด้วยการใช้ attribute "Style" กับ tags บางประเภทในภาษา HTML เพื่อกำหนดรูปแบบหน้าเว็บเพ็จให้เป็นไปตามที่ต้องการ ไม่ว่าจะเป็น รูปแบบของ ตัวอักษร (Type Face) สีพื้นของเอกสาร (Background) สีของตัวอักษร สีของลิงค์ (Link colors) ขนาดของขอบกระดาษ (Margin controls) และ กำหนดตำแหน่งของวัตถุ (Objects) บนหน้าเว็บเพ็จ

แม้ว่าเมื่อเริ่มต้นพัฒนาขึ้น ผู้พัฒนาภาษา HTML จะมีวัตถุประสงค์เพียงต้องการให้ HTML ทำหน้าที่เป็นภาษาที่ใช้จัดรูปแบบ เอกสารธรรมดาๆ โดยได้เตรียมเครื่องมือการจัดรูปแบบเบื้องต้นมาให้ไม่มากนัก แต่นักออกแบบเว็บเพ็จ (Web Designers) ซึ่งอยู่ในฐานะของผู้ใช้ภาษา ยังคงต้องการเครื่องมือเพื่อใช้ออกแบบเอกสารให้มีความสวยงาม มากกว่าที่จะใช้ภาษา HTML เพื่อจัดเก็บเอกสารรูปแบบธรรมดาแต่เพียงอย่างเดียว และด้วยเหตุผลดังกล่าว จึงเป็นแรงผลักดันให้ภาษา HTML เมื่อพัฒนามาถึงเวอร์ชัน 4 มีเครื่องมือในการจัดรูปแบบอย่างเพียบพร้อม

เพื่อเป็นการตอบสนองความต้องการดังกล่าว และยังคงวัตถุประสงค์ของภาษา HTML ในด้านการจัดเก็บเอกสาร ในเวอร์ชัน 4 นี้จึงได้พัฒนา Cascading Style Sheet (CSS) เพื่อเป็นเครื่องมือใช้ในการกำหนดรูปแบบเอกสาร และเมื่อมี Cascading Style Sheet เพื่อใช้กำหนด รูปแบบเอกสารแล้ว จึงได้กำหนดให้บาง tags ในเวอร์ชันก่อนๆ เช่น font tag เป็น tag ที่ถูกเลิกใช้แล้ว (Obsolete) ในเวอร์ชัน 4 นี้

หมายเหตุ: ยังคงมีหลายๆ web browser ที่ไม่ สนับสนุนการใช้ CSS อย่าง 100% อยู่พอสมควร แม้ว่าปัจจุบัน HTML เวอร์ชัน 4 จะถูกพัฒนามาหลายปีแล้วก็ตาม

- **ประเภทของ Style Sheets**

เราสามารถนำ Style Sheet มาใช้ในเอกสาร HTML ได้ 3 วิธีดังนี้

1. **Inline** เป็นวิธีกำหนดรูปแบบด้วยการใส่ Style attribute เข้ากับ tag วิธีนี้มีประสิทธิภาพสูงสุด เพราะสามารถกำหนด เฉพาะส่วนของเว็บเพ็จ ให้มีรูปแบบเฉพาะที่ต้องการได้ เช่น ต้องการให้ย่อหน้านั้นมีรูปแบบเฉพาะอย่าง ก็เพียงใส่ style="x" เข้ากับ Paragraph tag
2. **Embedded** เป็นวิธีกำหนดรูปแบบให้กับทั้งหน้าเว็บเพ็จ โดย การใช้ <STYLE> วางไว้ในส่วน <HEAD> ของหน้าเว็บเพ็จ แล้วเรียกใช้
3. **Linked** บางครั้งเรียก **External Style Sheet** เป็น CSS ที่เรียก ใช้ด้วยการเชื่อมโยง (Linked) เข้าสู่เอกสาร HTML มักใช้เพื่อเป็นแม่แบบให้กับเอกสารหลายๆ หน้า เราสร้าง CSS โดยใช้ Text editor เขียนคำสั่งเพื่อกำหนดรูปแบบที่ต้องการ เมื่อเสร็จ save ไฟล์ให้มีนามสกุล .css โดยวางไฟล์นี้ ไว้ใน folder เดียวกันกับเว็บเพ็จที่จะ linked เราสามารถ linked เว็บเพ็จเข้ากับ CSS ประเภทนี้ได้โดยไม่จำกัดจำนวนหน้า นั่นหมายความว่า ไม่มีความแตกต่างระหว่างเว็บเพ็จจำนวน 1-2 หน้า กับจำนวนเป็นพันๆ หน้าที่จะ linked เข้าใช้กับ CSS ประเภทนี้

ต่อจากนี้ เราจะทำความเข้าใจกับการใช้ CSS ในแต่ละแบบ โดยพิจารณาจากต.ย. เป็นประเภทๆ ไป แต่คุณบางคนอาจจะงงๆ หากยัง ไม่คุ้นกับ CSS อย่างไรก็ตาม สิ่งที่เรียนรู้จากต.ย. ในเบื้องต้น ก็คือวิธีการกำหนดและเรียกใช้ CSS และหลังจากนั้น เราจะพิจารณารูปแบบ (Syntaxes) ของ CSS กัน

- **Inline Style**

ตัวอย่าง การใช้ Inline Style

```
<HTML>
<HEAD>
<TITLE>การใช้ Inline Cascading Style Sheet</TITLE>
</HEAD>
<BODY>
<P>
รูปแบบและขนาดของตัวอักษรในย่อหน้านี้ จะมีขนาดและรูปแบบตัวอักษรเป็นไปตามค่า default ของ web
browser
</P>
<P STYLE="font: 16pt BrowaliaUPC;">
รูปแบบและขนาดของตัวอักษรในย่อหน้านี้ จะมีขนาด 16 point และรูปแบบตัวอักษรเป็น BrowaliaUPC
</P>
</BODY>
</HTML>
```

การใช้ Inline Style Sheet เหมาะกับ งานที่กำหนดคุณสมบัติของแต่ละส่วนบนเว็บเพ็จ ให้มีรูปแบบเฉพาะตัว ด้วยเหตุนี้ เราจึงมี tags 2 ประเภทเข้ามาเกี่ยวข้อง ได้แก่ <DIV> และ <SPAN> Tags ทั้งสอง มีบทบาทเพราะ

ทำหน้าที่กำหนดรูปแบบของเว็บเพ็จเป็นส่วนๆ Tags ที่อยู่ระหว่าง tags ทั้งสองจะถูกกำหนดให้มีรูปแบบและลักษณะตาม tags ทั้งสองโดยอัตโนมัติ นอกจากนี้ Tags ทั้งสอง

ความแตกต่างของ <DIV> กับ <SPAN> อยู่ที่ <DIV> จะใส่ line break ให้หลังจบ </DIV> เพื่อให้ขึ้นบรรทัดใหม่ ในขณะที่ <SPAN>...</SPAN> จะไม่

ตัวอย่าง การใช้ Inline Style

```
<HTML>
<HEAD>
<TITLE>การใช้ Inline Cascading Style Sheet</TITLE>
</HEAD>
<BODY>
<P>
รูปแบบและขนาดของตัวอักษรในย่อหน้านี้ <DIV STYLE="font: 12pt BrowaliaUPC; color: #999999";>
จะมีขนาด 12 point </DIV>และรูปแบบตัวอักษรเป็น BrowaliaUPC
</P>
<P STYLE="font: 12pt BrowaliaUPC;">
รูปแบบและขนาดของตัวอักษรในย่อหน้านี้ <SPAN STYLE="font: 10pt BrowaliaUPC; color:
#999999";> จะมีขนาด 12 point </SPAN> และรูปแบบตัวอักษรเป็น BrowaliaUPC
</P>
</BODY>
</HTML>
```

- **Embedded Style**

ตัวอย่าง การใช้ Embedded Style

```
<HTML>
<HEAD>

<TITLE>กำหนดรูปแบบด้วย Embedded Style Sheet</TITLE>
<STYLE>
<--
BODY {
background: #FFFFFF;
color: #000000;
}
H3 {
font: 14pt BrowaliaUPC; color: #CCCCCC;
}
P {
```

```
font: 12pt AngsanaUPC;
}
A {
color: #FF0000; text-decoration: none;
}
-->
</STYLE>
</HEAD>

<BODY>
<H3>ประเภทของ Style Sheets</H3>
<P>
เราสามารถนำ Style Sheet มาใช้ในเอกสาร HTML ได้ 3 วิธีดังนี้<br>
Inline เป็นวิธีกำหนดรูปแบบด้วยการใส่ Style attribute เข้ากับ tag วิธีนี้มีประสิทธิภาพสูงสุด เพราะสามารถกำหนดให้เว็บเพ็จเฉพาะส่วน ที่ต้องการ ให้มีรูปแบบเฉพาะที่แตกต่างไปจาก ส่วนอื่นๆ บนเว็บเพ็จได้ เช่น ต้องการให้ย่อหน้านั้นมีรูปแบบเฉพาะ ก็เพียงใส่
<A HREF="#style">style="x" </A>เข้ากับ Paragraph tag ก็เป็นที่เรียบร้อย
</P>
</BODY>

</HTML>
```

จากตัวอย่างข้างต้น คงจะสังเกตเห็นว่า โครงสร้าง ของ CSS แตกต่างไปจากรูปแบบทั่วๆ ไปของ HTML แต่ลักษณะการเขียน ยังคงไม่แตกต่าง กล่าวคือ เรากำหนดสีพื้นของเว็บเพ็จให้มีสีขาว สีตัวอักษรสีดำ กำหนดรูปแบบตัวอักษร ขนาด ของ heading และ Paragraph ตลอดจนสีของลิงค์ ตามลำดับ

หมายเหตุ:

1. เรากำหนดขนาดของตัวอักษรในหน่วยของ point (pt), pixels (px), percent (เช่น 75%) หรือเซนติเมตร ได้
2. นักเขียนเว็บเพ็จบางท่าน นิยมจะกำหนด Embedded CSS ด้วยการใส่เครื่องหมาย Comment (<!-- >) ล้อมรอบ code ของ CSS เพื่อซ่อนคำสั่งของ CSS ออกจาก web browser ที่ไม่สนับสนุนการใช้ CSS เช่น <!-- Style Sheet codes goes here -->

- **Linked Style**

Linked Style Sheet หรือ External Style Sheet ใช้โครงสร้างภาษาเดียวกันกับ Embedded CSS เพียงแต่ส่วนของ code ถูกแยกออกเป็นไฟล์ต่างหากจากเอกสาร HTML

ตัวอย่าง การใช้ Linked Style Sheet

```
<STYLE>
 BODY {
```

```
background: #FFFFFF;
color: #000000;
}
H3 {
font: 14pt BrowaliaUPC; color: #CCCCCC;
}
P {
font: 12pt AngsanaUPC;
}
A {
color: #FF0000; text-decoration: none;
}
-->
</STYLE>
```

เขียนคำสั่งข้างต้นด้วยโปรแกรม Text editior เช่น Notepad เมื่อเสร็จ Save ไฟล์นี้ไว้ใน folder เดียวกันกับหน้าเว็บเพ็จอื่นๆ โดยตั้งชื่อที่เหมาะสม แต่จะต้องมีนามสกุล (Extension) เป็น .css เท่านั้น ในกรณีนี้ให้ใช้ชื่อเป็น style.css

นำตัวอย่าง Embedded Style กลับมาปรับปรุงเล็กน้อยดังนี้

```
<HTML>
<HEAD>
 <TITLE>กำหนดรูปแบบด้วย Linked Style Sheet</TITLE>
<LINK rel=stylesheet href="style.css" type="text/css">
</HEAD>
 <BODY>
<H3>ประเภทของ Style Sheets</H3>
<P>
เราสามารถนำ Style Sheet มาใช้ในเอกสาร HTML ได้ 3 วิธีดังนี้<br>
Inline เป็นวิธีกำหนดรูปแบบด้วยการใส่ Style attribute เข้ากับ tag วิธีนี้มีประสิทธิภาพสูงสุด เพราะสามารถกำหนดให้เว็บเพ็จเฉพาะส่วน ที่ต้องการ ให้มีรูปแบบเฉพาะที่แตกต่างไปจาก ส่วนอื่นๆ บนเว็บเพ็จได้ เช่น ต้องการให้ย่อหน้านั้นมีรูปแบบเฉพาะ ก็เพียงใส่
<A HREF="#style">style="x" </A>เข้ากับ Paragraph tag ก็เป็นที่เรียบร้อย
</P>
</BODY>
 </HTML>
```

ผลลัพธ์จะได้เหมือนกันกับตัวอย่าง Embedded Style แต่สิ่งที่แตกต่าง คือ การกำหนดด้วย Embedded CSS สามารถใช้กับเว็บเพ็จหน้านั้นหน้าเดียว หน้าอื่นจะมาเรียกใช้ไม่ได้ ในขณะที่ Linked CSS กำหนด ไว้เป็น

ไฟล์แยกต่างหากจากเว็บเพ็จ เว็บเพ็จหน้าที่ต้องการใช้ CSS จะเรียกใช้ได้ด้วยการใช้ <LINK> ซึ่งเขียนไว้ในส่วนหัวของเว็บเพ็จ (Head section) หน้านั้นๆ

- **นำ CSS ทั้ง 3 แบบมาใช้ร่วมกัน**

เราสามารถนำ CSS ทั้ง 3 ประเภทมาใช้ร่วมกันได้ สมมติว่าเรากำลังจะใช้ CSS กับเว็บไซต์ขนาดใหญ่สักเว็บไซต์หนึ่ง เราสามารถใช้ Linked CSS เพียงหนึ่งไฟล์ ควบคุมเว็บเพ็จทุกๆ หน้าบนเว็บไซต์ได้ และใช้ Embedded CSS และ/หรือ Inline CSS เพื่อปรับแต่ง รายละเอียดเล็กๆ น้อยๆ เพื่อให้เว็บเพ็จหน้านั้นๆ มีลักษณะที่แตกต่างไปจากทุกๆ หน้าที่ถูกควบคุมด้วย Linked CSS ได้

เมื่อนำ CSS ทั้งสามแบบมาใช้ร่วมกัน โปรแกรม web browser จะจดจำลักษณะตาม Linked CSS เป็นลำดับแรก แล้วจึงมองหา ลักษณะ เพิ่มเติมจาก Embedded และ Inline CSS ตามลำดับ แล้วจึงกำหนดค่าให้กับ tag ที่เกี่ยวข้อง ถ้าหากมีการกำหนด ลักษณะซ้ำด้วย Embedded และ/หรือ Inline CSS ให้กับ tag ที่ได้กำหนดไว้แล้วใน Linked CSS โปรแกรม web browser จะ override ค่าที่กำหนด โดย Linked CSS ด้วย Embedded และ/หรือ Inline CSS ตามลำดับ

นอกจากนี้ เราสามารถใช้ Inline CSS กำหนดลักษณะของ tag เพื่อ override ค่าที่กำหนดด้วย Embedded CSS ได้ โดยที่ web browser ยังคงมี priority ในการทำงานโดยไล่จาก Linked CSS, Embedded CSS และ Inline CSS ตามลำดับ

เราสร้าง style.css ไว้แล้ว เราจะนำมาใช้กับตัวอย่างนี้

ตัวอย่าง

```
<HTML>
<HEAD>

<TITLE>ลำดับการทำงานของ Cascading Style Sheet</TITLE>
<LINK rel=stylesheet href="style.css" type="text/css">
<STYLE>
<!--
P {
font: 12pt CordiaUPC;
color: #000088;
}
-->
</STYLE>
</HEAD>

<BODY>
<H3 style="font: 12pt MS Sans Serif; line-height: 16pt; color: #999999;">Cascading Style Sheet (1)</H3>
<P>
แม้ว่าภาษา HTML เวอร์ชันก่อนหน้านี้ จะมีเครื่องมือในการจัดรูปแบบให้มาพร้อม แต่การนำมาใช้ก็
```

```
ค่อนข้างยุ่งยาก และเยิ่นเย้อ เพราะกว่าจะกำหนดรูปแบบที่ต้องการได้ ก็ต้องใช้ tags หลายๆ ประเภทซ้อน
กันจนเป็นแถวยาวเหยียด เวลา debug ก็หากันตา(แทบ) เหล่ กว่าจะเจอที่ผิด
</P>
<P>
อย่างไรก็ตาม ปัญหาดังกล่าวจะเบาบางลง (แม้ว่าจะไม่หมดไป) เมื่อนำ <A
HREF="html16.html">Cascading Style Sheet</A> มาใช้ ทำให้ชีวิตของนักเขียนเว็บเพ็จอย่างเราๆ ท่านๆ
สบายขึ้นบ้าง
</P>
</BODY>


</HTML>
```

- **Class**

Class เป็นวิธีกำหนดลักษณะเฉพาะให้กับข้อความ เพื่อที่ข้อความนั้นจะแตกต่างไปจากข้อความส่วนอื่นๆ วิธีการนี้เรียกว่า Style class

สมมติว่า เราต้องการให้มีบางย่อหน้า มีลักษณะ ที่แตกต่างกัน เราทำได้โดยกำหนดค่าต่อไปนี้ลงในไฟล์ CSS ดังนี้

```
<STYLE>
<!--
P.verdana{
font: 10pt Verdana;
color: #999999;
}
P.arial {
font: 8pt Arial;
color: #000000;
}
-->
</STYLE>
```

การใช้งาน Class

เรากำหนด class ด้วยการตั้งชื่อให้กับ tag ที่ เราต้องการ โดยเพิ่ม . (period) แล้วตามด้วยชื่อ แล้วเรียกใช้ด้วย tag นั้น ตามด้วย style="x" เมื่อ x เป็นชื่อที่กำหนด พิจารณาตัวอย่างต่อไปนี้

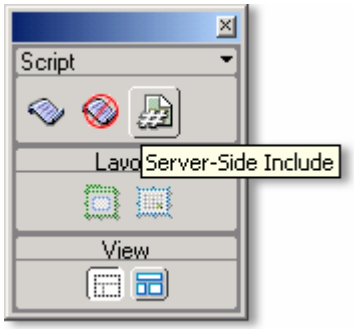
```
<HTML>
<HEAD>


<TITLE>Style sheet sample: class</TITLE>
<STYLE>
```

```
<!--
P.center {
font-family: garamond, times, serif;
font-size: 10pt;
text-align: center;
}
P.right {
font-family: garamond, times, serif;
font-size: 8pt;
text-align: right;
}
P.name {
font-family: garamond, times, serif;
font-size: 8pt;
text-align: center;
text-weight: bold;
text-style: italic;
}
-->
</STYLE>
</HEAD>

<BODY>
<P class="center">
Let my first words in this book be an apology for whatever errors do appear here, and for whatever overlooked toics don't. I have done my darnedest to make this the most useful book possible, the book that I would have wanted when I began digging past the IBM/PC's easy surface soil into its dark and fascinating subterranean reaches.
</P>
<P class="right">
The author and publisher of this book have used their best efforts in preparing this book and the programs contained in it. These efforts include the development, research, and testing of the theories and programs to determine their effectiveness. The author and publisher make no warranty of any kind, expressed or implied, with regard to these programs or the documentation contained in this book. The author and publisher shall not be liable in any even for incidental or consequential damages in connection with, or arising out of, the furnishing, performance, or use of these programs.
</P>
<P class="name">
```

```
-- Michael B. Little
</P>
</BODY>

</HTML>
```

หลักการ Include

การ include คือการเรียกใช้ file จากภายใน Homepage เป็นการลดความซ้ำซ้อน การใช้ include คุณสามารถเรียก file ได้ไม่จำกัด (สะดวกกับการใช้ติดต่อกับฐานข้อมูล) แต่ถ้าใช้กับ การออกแบบสร้าง Homepage แล้ว คุณไม่จำเป็นต้องศึกษาภาษา Script เหล่านั้น แต่เลือกใช้ Tool ที่มี Function include แล้ว Save เป็น นามสกุล ของ Script เหล่านั้น เช่น .asp , .php เป็นต้น



การ include ประกอบด้วย file มากกว่า 2 ขึ้นไป

1. fileที่ 1 คือ file ที่เปิดผ่าน browser
2. fileที่ 2 คือ fileที่ถูกเรียกมาจาก fileที่ 1 (Server-Side-include)

คำสั่ง include ใน ASP

<!--#include virtual="/path/file.htm"-->

เรียก file จากด้านนอก เริ่มต้นจาก root Directory (\root\path\file.htm)

<!--#include file="../path/file.htm"-->

เรียก file จากด้านใน ย้อนออกมาเข้าใน Directory ที่อยู่ของ file ที่เรียกใช้ (Parent Path)

หรือ เรียก file ใน Directory เดียวกันมาใช้ เช่น <!--#include file="file.htm"-->

ตัวอย่าง ASP Template

```
<!--#include file="Top.htm"-->
<!--#include file="Head.htm"-->
<table width="100%" border="0">
<tr>
<td><!--#include file="Content.htm"--></td>
```

```
<td>ข้อความใน HTML </td>
</tr>
</table>
<!--#include file="Bottom.htm"-->
```

คำสั่ง include ใน PHP

```
<?
include("../path/file.htm");
?>
```

ตัวอย่าง PHP Template

```
<?
include("Top.htm");
include("Head.htm");
?>
<table width="100%" border="0">
<tr>
<td><? include("Content.htm"); ?></td>
<td>ข้อความใน HTML </td>
</tr>
</table>
<?
include("Bottom.htm");
?>
```