

การแสดงผล Theme ด้วย GUI

- การแสดงผล Theme ด้วย GUI

ดัดแปลงและเรียบเรียงจาก หนังสือ Introduction to AVENUE โดย ESRI

โดย... อาจารย์สุเพชร จิระจรรกุล

ในบทความชุดนี้ทางผู้เรียบเรียงและเขียน ได้อธิบายวิธีการแสดงผล Theme ที่เราสนใจข้อมูลนั้น ซึ่งในบทความนี้อยากให้ท่านได้เรียนรู้โครงสร้างของชุดคำสั่งในการเรียกใช้งาน theme และจะได้เข้าใจถึงโครงสร้างของโปรแกรมแบบ For Each ... มากขึ้น ในการเรียกใช้งานซ้ำๆ สำหรับบางกรณี

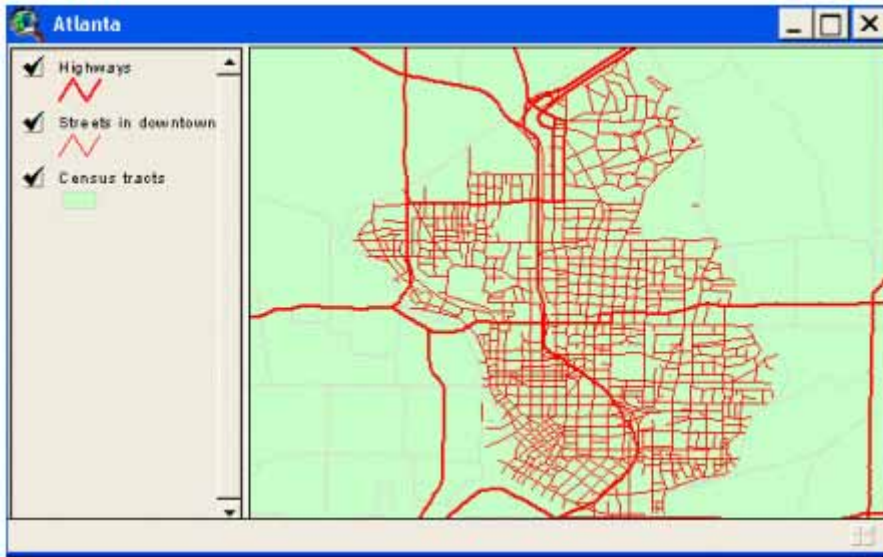
ให้ทุกท่านเปิด Project ที่ทำไว้ในบทความครั้งที่ 4 ที่ผ่านมาแล้วให้ลองทดสอบ script ใหม่ เพื่อให้เราเข้าใจในบทเรียนมากขึ้น

ซึ่งถ้าเราพยายามทำความเข้าใจโครงสร้างของชุดคำสั่งต่างๆ ที่เราต้องการเรียกใช้งานให้ Theme ที่มีอยู่แสดงผล เราอาจจะใช้คำสั่ง SetVisible ดังตัวอย่างข้างล่างเป็นการค้นหา Theme ที่ต้องการ และให้มีการแสดงผลบน view

```
theView = av.GetProject.FindDoc("Atlanta")
```

```
theTheme = theView.FindTheme("Highways")
```

```
theTheme.SetVisible(true)
```

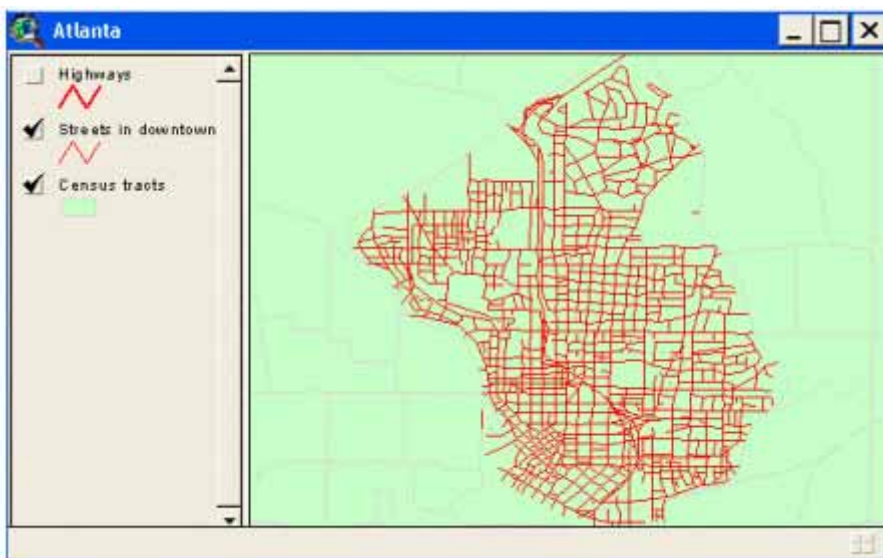


แต่ถ้าเราต้องการเลือก Theme นั้นและยกเลิกการแสดงผลบน View ดังนี้

```
theView = av.GetProject.FindDoc("Atlanta")
```

```
theTheme = theView.FindTheme("Highways")
```

```
theTheme.SetVisible(false)
```



นั่นคือโครงสร้างของชุดคำสั่งในการแสดงผลที่เราต้องการเลือกแสดงหรือไม่แสดงนั้น เราจะต้องทำการสร้างตัวแปรขึ้นมาในที่นี้คือ theView เพื่อทำการเก็บค่า view ที่สนใจไว้โดยผ่านตัวแปรเก็บค่า application คือ av แล้วตามด้วยค่าโปรเจกต์ปัจจุบันโดยผ่านตัว request ชื่อ GetProject

แล้วให้ทำการค้นหา View ที่สนใจโดยผ่าน request คือ FindDoc นั้นเอง

ตัวแปรตัวต่อไปที่เราตั้งขึ้นมาคือ theTheme เพื่อเป็นตัวเก็บค่า Theme ที่เราสนใจในที่นี้ชื่อ Highways เราก็ต้องใช้ตัวแปรที่อ้างอิงมาก่อนหน้าคือ theView แล้วให้ค้นหา Theme ที่สนใจ โดยผ่าน request ชื่อ FindTheme ถ้าเจอก็จะได้รับค่ามาผ่าน theTheme จากนั้นให้นำค่า อ้างอิงข้างต้นมาใส่ request การแสดงผลคือ SetVisibleว่าจะให้ แสดง (true) หรือ ไม่แสดง (false) บน View ดังกล่าว

แต่ในบางครั้งเราไม่อยากจะมานั่งใส่ค่า request ของ Theme เป็น SetVisible(true) หรือ SetVisible(false) เราอาจจะใช้รูปแบบโครงสร้างของชุดคำสั่งการเปิดปิด Theme โดยใช้ if...then...Else ก็ได้ เพื่อเป็นการกำหนดเงื่อนไขที่ง่ายมากขึ้นกว่าเดิม โดยให้โปรแกรมควบคุมการแสดงผลเปิดปิดโดยสลับกันโดยมีการตรวจสอบเงื่อนไขว่า ถ้าเปิดอยู่ ให้ปิด และถ้าปิดอยู่ให้เปิด ให้พยายามสังเกตในเงื่อนไข

if (conditionที่เป็นจริง) then

‘เป็นจริงทำงานที่ต้องการ-1

else

‘ถ้าเป็นเท็จทำงานที่ต้องการ-2

end

ใน (condition) หรือเงื่อนไขต้องอยู่ภายในวงเล็บเท่านั้น

จากโครงสร้างชุดคำสั่งข้างบนให้นำมาดัดแปลงดังนี้

```
theView = av.GetProject.FindDoc("Atlanta")
```

```
theTheme = theView.FindTheme("Highways")
```

```
if (theTheme.IsVisible) then
```

```
theTheme.SetVisible(false)
```

```
else
```

```
theTheme.SetVisible(true)
```

```
end
```

สังเกตว่าในการตรวจสอบค่าการแสดงผลโดยใช้ request ชื่อ IsVisible นั้นสามารถทราบได้ว่า Theme เปิดหรือปิดอยู่

ส่วนข้อสังเกต 2 บรรทัดบน ในกรณีที่ View ชื่อ Atlanta นั้น Active อยู่ เราอาจจะรวม 2 คำสั่งข้างต้นให้เป็นบรรทัดเดียวโดยใช้ request ชื่อ GetActiveDoc หมายถึงให้เก็บค่า Document (View, Table, Layout, Script) ที่กำลัง Active หรือทำงานอยู่ ดังนั้นการใช้งานจึงต้องระมัดระวังไม่เช่นนั้นอาจจะเกิดความผิดพลาดในการสั่งการทำงานได้ ซึ่งถ้ารวมกันก็จะได้ดังชุดคำสั่งข้างล่างครับ

```
theTheme = av.GetActiveDoc.FindTheme("Highways")
```

```
if (theTheme.IsVisible) then
```

```
theTheme.SetVisible(false)
```

```
else
```

```
theTheme.SetVisible(true)
```

```
end
```

แต่ถ้าใช้ทดสอบโปรแกรมนั้นในชุดคำสั่งที่สองนี้จะต้องสร้างเป็น ปุ่มหรือ เมนูบาร์ไว้ครับแล้วทำ View ที่สนใจให้ Active คำสั่งจึงประมวผลได้ไม่ผิดพลาด

ปัญหาต่อไปคือในทางปฏิบัติเราอยากให้โปรแกรมทำงานโดยต้องตรวจสอบว่า มี Theme อื่นๆใหม่ที่ น่าสนใจนอกเหนือไปจาก Highways ให้โปรแกรมช่วยแสดงออกมาให้เราเลือกใช้งานด้วย เพื่อเป็นการเพิ่มความสะดวกในการทำงานมากขึ้นครับ เพราะคำสั่งข้างต้นนั้นทำได้เฉพาะ Theme ที่ชื่อ Highways เท่านั้นครับ

ซึ่งคำสั่งในการแสดงผลว่ามี Theme ไດบ้างใน View ต้องใช้ request ชื่อ GetThemes

เพื่อให้รับค่าคือชื่อ Theme ต่างๆ จาก View ให้ User สามารถเลือกใช้งานได้

โดยจากชุดคำสั่งข้างต้นให้ดัดแปลงเป็นชุดคำสั่งใหม่ดังนี้

```
theView = av.GetProject.FindDoc("Atlanta")
```

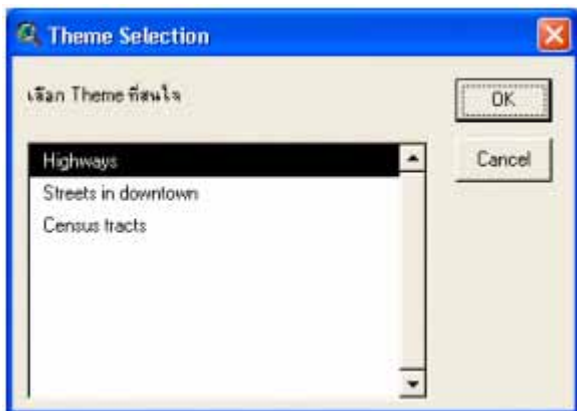
```
theThemeList = theView.GetThemes
```

จากนั้นให้เราใส่ค่าแสดงรายการชื่อ Theme ต่างๆ ออกมาโดยใช้ Message Box มาช่วยในขั้นตอนนี้โดยใช้ชุดคำสั่งให้เพิ่มเป็นดังนี้

```
theView = av.GetProject.FindDoc("Atlanta")
```

```
theThemeList = theView.GetThemes
```

```
theTheme = MsgBox.List(theThemeList, "เลือก Theme ที่สนใจ","Theme Selection")
```



จากนี้เราจะเริ่มดัดแปลงให้รับค่าชื่อ Theme ที่ผู้ใช้จะเลือกผ่าน Message Box ใน List รายการ เพียง 1 รายการข้างต้น โดยใช้รูปแบบคำสั่ง MsgBox.List หรือ MsgBox.Choice ก็ได้ แล้วนำไปเปิดหรือปิด Theme ดังนี้

```
theView = av.GetProject.FindDoc("Atlanta")
```

```
theThemeList = theView.GetThemes
```

```
theTheme = MsgBox.List(theThemeList, "เลือก Theme ที่สนใจ","Theme
```

```

Selection")

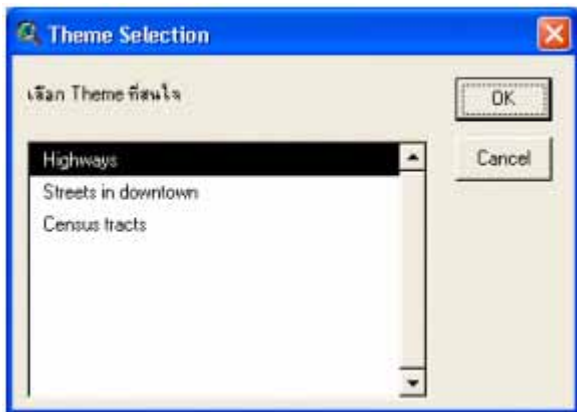
if (theTheme.IsVisible) then

theTheme.SetVisible(false)

else

theTheme.SetVisible(true)

end
    
```



หรือใช้รูปแบบคำสั่ง MsgBox.Choice ... ดังนี้

```

theView = av.GetProject.FindDoc("Atlanta")

theThemeList = theView.GetThemes

theTheme = MsgBox.Choice(theThemeList, "เลือก Theme ที่สนใจ","Theme
Selection")

if (theTheme.IsVisible) then

theTheme.SetVisible(false)

else

theTheme.SetVisible(true)
    
```

end



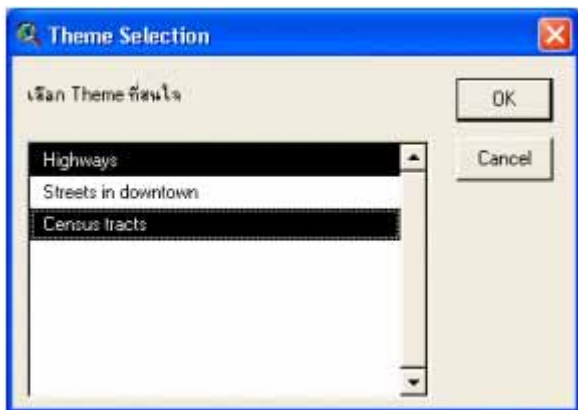
ในขั้นนี้เราสามารถควบคุม Theme ให้แสดงผลเปิดและปิด ตาม List ที่ผู้ใช้เลือกได้เพียง 1 Theme เท่านั้น ได้แล้วครับ

เมื่อใดที่ต้องการแสดงผล Theme มากกว่า 1 รายการให้ใช้ MsgBox.MultiList ดังนี้

```
theView = av.GetProject.FindDoc("Atlanta")
```

```
theThemeList = theView.GetThemes
```

```
theTheme = MsgBox.MultiList(theThemeList, "เลือก Theme ที่สนใจ","Theme Selection")
```



แต่เนื่องจากผู้ใช้อาจจะเลือกมากกว่า 1 Theme ดังนั้นใช้ if...then..else แบบเดิมเพียงอย่างเดียวไม่ได้แล้ว เราจะต้องเปลี่ยนรูปแบบไปใช้ for each... loop ผสมเข้ามา ซึ่งมีโครงสร้างดังนี้

```
for each วัตถุ in รายการที่เก็บค่าวัตถุ
```

ทำตามคำสั่งตามต้องการ

end

ซึ่งเราจะต้องดัดแปลงให้เป็นไปตามชุดคำสั่งดังนี้

```
theView = av.GetProject.FindDoc("Atlanta")
```

```
theThemeList = theView.GetThemes
```

```
theTheme = MsgBox.MultiList(theThemeList, "เลือก Theme ที่สนใจ","Theme Selection")
```

```
for each thms in theTheme
```

```
if (thms.IsVisible) then
```

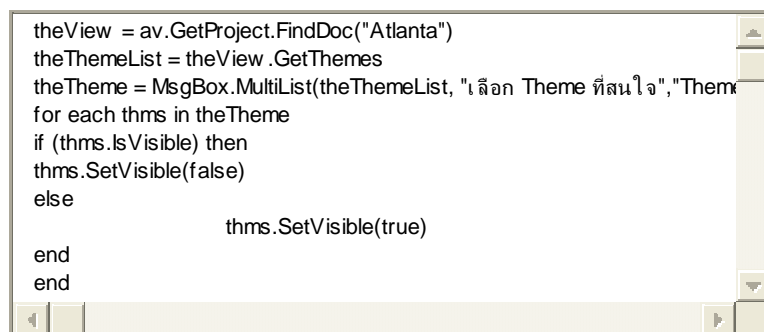
```
    thms.SetVisible(false)
```

```
else
```

```
    thms.SetVisible(true)
```

```
end
```

```
end
```



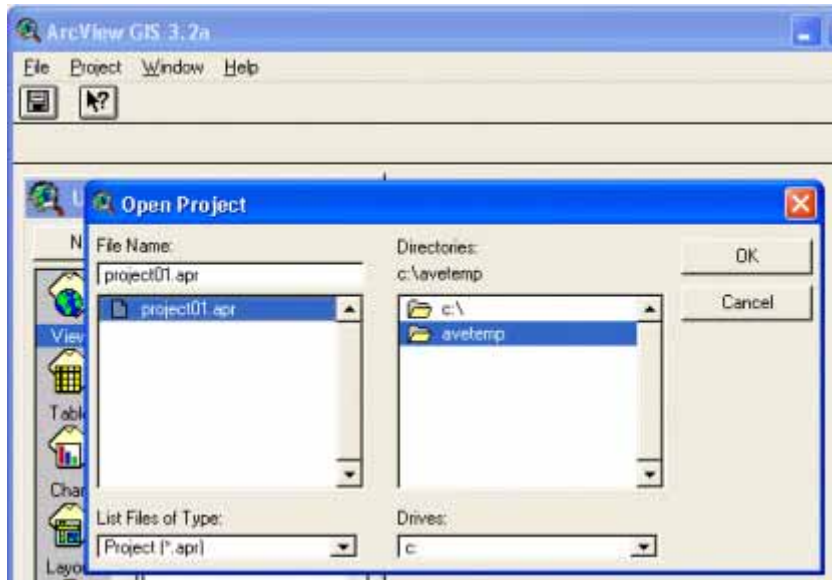
```

theView = av.GetProject.FindDoc("Atlanta")
theThemeList = theView .GetThemes
theTheme = MsgBox.MultiList(theThemeList, "เลือก Theme ที่สนใจ","Theme Selection")
for each thms in theTheme
if (thms.IsVisible) then
thms.SetVisible(false)
else
    thms.SetVisible(true)
end
end
    
```

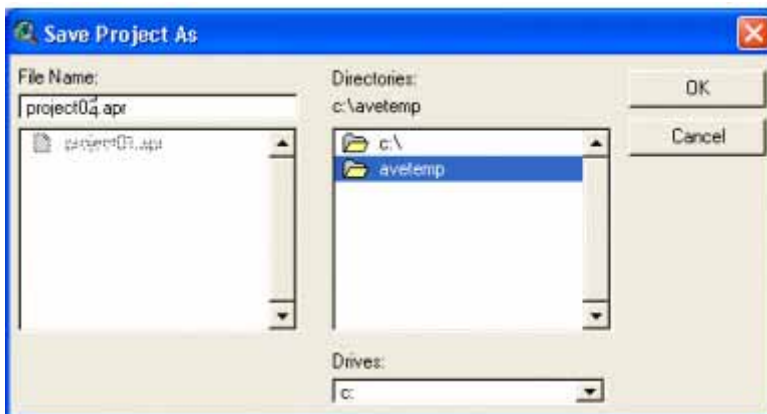
ขั้นตอนต่อไปเราก็จะลองทำแบบฝึกหัดต่อจากคราวที่ผ่านมา โดยให้ทำตามขั้นตอนดังนี้

ขั้นที่ 1 ทำการเปิดแฟ้มโปรเจคและบันทึกข้อมูล Project ใหม่

เพื่อไม่ให้ข้อมูล Project เดิมเสียหายไปจากการปรับแต่ง ดังนั้นผู้เขียน GUI จะต้องทำการเปิดแฟ้มโปรเจกของคราวที่ผ่านมาโดยเลือกที่เมนู File - Open Project... แล้วให้เลือกไปที่ C:\AVETEMP แล้วเลือกไฟล์ชื่อ Project01.apr



ทำการบันทึกไปเป็นแฟ้มข้อมูลใหม่โดยเลือกที่เมนู File - Save Project As... แล้วให้เลือกไปที่ C:\AVETEMP นั้นเอง แล้วตั้งชื่อไฟล์ว่า Project02.apr



ขั้นที่ 2 การสร้างชุดคำสั่งเพื่อเปิดและปิด Theme

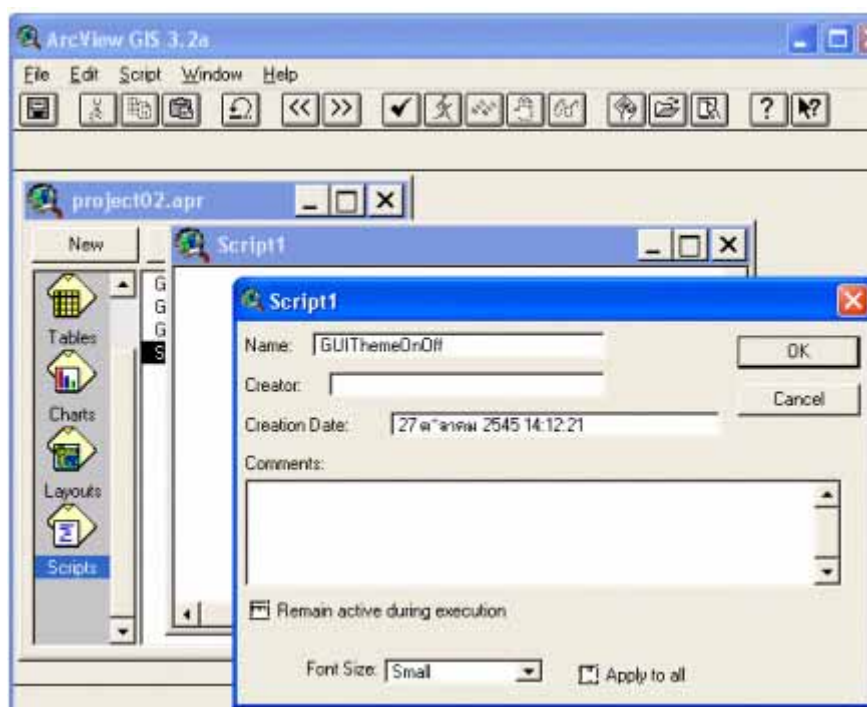
โดยให้เขียนชุดคำสั่งดังนี้ เนื่องจากครั้งที่ผ่านมามี View ที่ทำไว้แล้วถึง 3 View ดังนั้นในการเรียกแต่ละ View อาจจะมี Theme ที่ไม่เหมือนกัน หรือชื่อไม่ตรงกัน ดังนั้นเราจึงควรให้โปรแกรมทำการตรวจสอบและนำข้อมูลรายชื่อ Theme มาแสดงให้เราเลือกใช้งาน โดยเราจะนำไปทำเป็น ปุ่ม หรือเมนูคำสั่ง บน View Window โดยดัดแปลงชุดคำสั่ง

ให้คุณเลือกไปที่ Project Window

จากนั้นเลือกที่ icon ชื่อ Scripts แล้วกดปุ่ม New เราจะได้ Script 1 ขึ้นมา

จากนั้นให้เลือกที่คำสั่งเมนู Script - Properties... แล้วเปลี่ยน Name เป็น GUIThemeOnOff

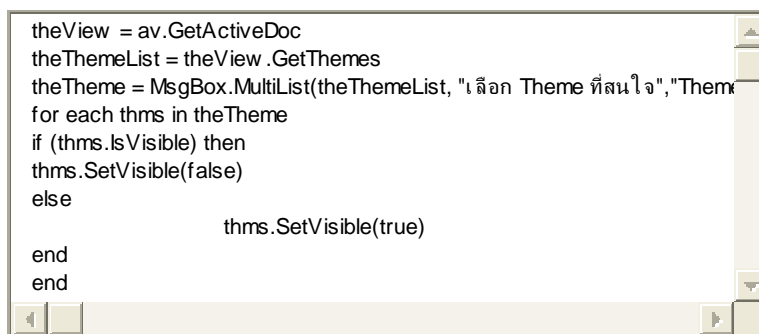
แล้วกดปุ่ม OK มาดังรูป



แล้วให้ทำการพิมพ์คำสั่งใน Script Editor ดังนี้ (นั่นคือเรานำคำสั่งเดิมข้างบนที่ได้อธิบายไว้แล้ว โดยละเอียดมาดัดแปลงให้เหมาะสมในการเรียกเปิดปิด Theme ซึ่งจะสามารถเรียกได้ทุกๆ View ที่ถูกเปิดอยู่ หรือ Active อยู่นั่นเอง พิมพ์ดังนี้

```
theView = av.GetActiveDoc  
  
theThemeList = theView.GetThemes  
  
theTheme = MsgBox.MultiList(theThemeList, "เลือก Theme ที่สนใจ","Theme  
Selection")  
  
for each thms in theTheme  
  
if (thms.IsVisible) then  
  
        thms.SetVisible(false)  
  
else  
  
thms.SetVisible(true)  
  
end  
  
end
```

หรือคัดลอกจาก textbox ข้างล่าง



```
theView = av.GetActiveDoc  
theThemeList = theView .GetThemes  
theTheme = MsgBox.MultiList(theThemeList, "เลือก Theme ที่สนใจ","Them  
for each thms in theTheme  
if (thms.IsVisible) then  
thms.SetVisible(false)  
else  
        thms.SetVisible(true)  
end  
end
```

จากนั้นให้กดปุ่ม compile เพื่อแปลภาษา เพื่อตรวจสอบ Syntax error ว่ามีหรือไม่



ถ้าไม่มี error ในด้านโครงสร้างและคำสั่งของโปรแกรม

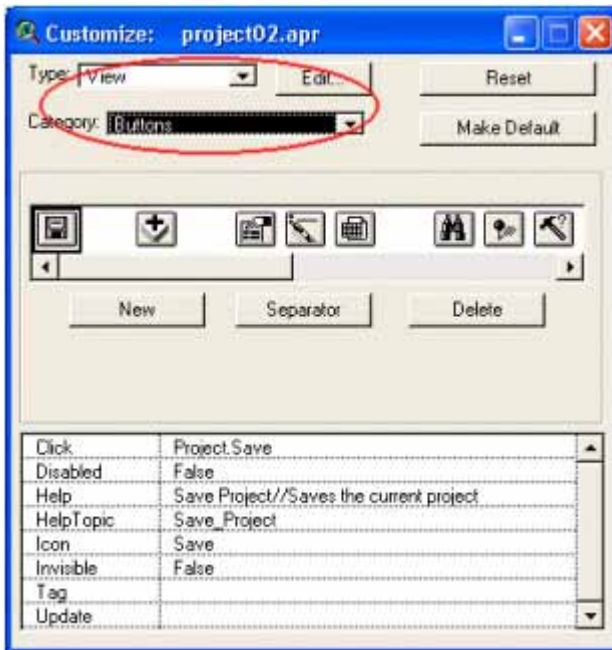
จากนั้นให้เรา ปิด Script Editor ออกไป

ตอนนี้จะอยู่ที่ Project Window เหมือนเดิม

ให้ Double Click ที่ Button Bar หรือ Tool bar เพื่อเรียก Customize



ใน Customize : Dialog ให้เลือกที่แถบ Type: ให้เป็น View และ ในส่วนของ Category ให้เป็น Buttons



แล้วในส่วนของ Control Editor หรือรูปปุ่มนั้นให้เลื่อนไปท้ายสุด แล้วเลือกที่ปุ่มท้ายสุด

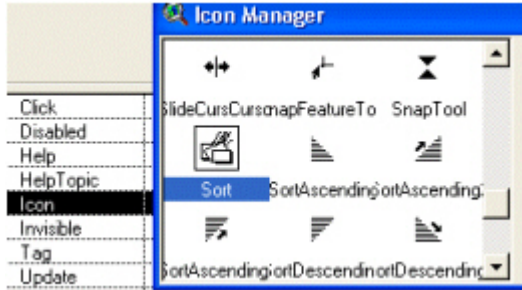
จากนั้นกดปุ่ม Separator 1 ครั้ง

กดปุ่ม New 1 ครั้งเพื่อสร้างปุ่มใหม่

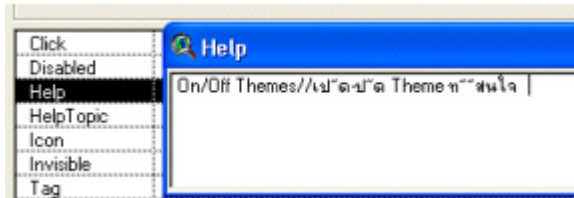


จากนั้นให้ทำการปรับแต่งส่วนของ Properties List ดังนี้

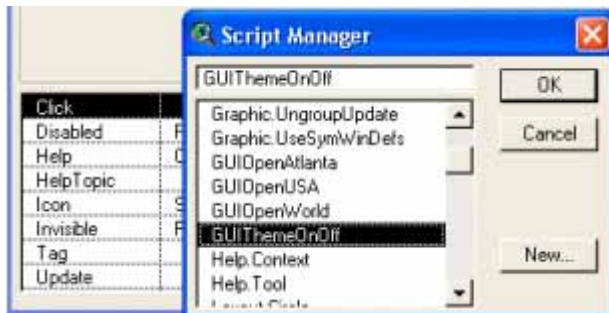
double click ที่แถบชื่อ icon เพื่อเลือกรูปของปุ่ม ในที่นี้ตัวอย่างใช้ Sort กดปุ่ม OK



Double click ที่แถบ Help แล้วพิมพ์ข้อความว่า "On/Off Themes//เปิด-ปิด Theme ที่สนใจ" กดปุ่ม OK



Double click ที่แถบ Click แล้วเลือกชุดคำสั่งที่ตั้งชื่อไว้เมื่อสักครู่ที่ว่า "GUIThemeOnOff" กดปุ่ม OK



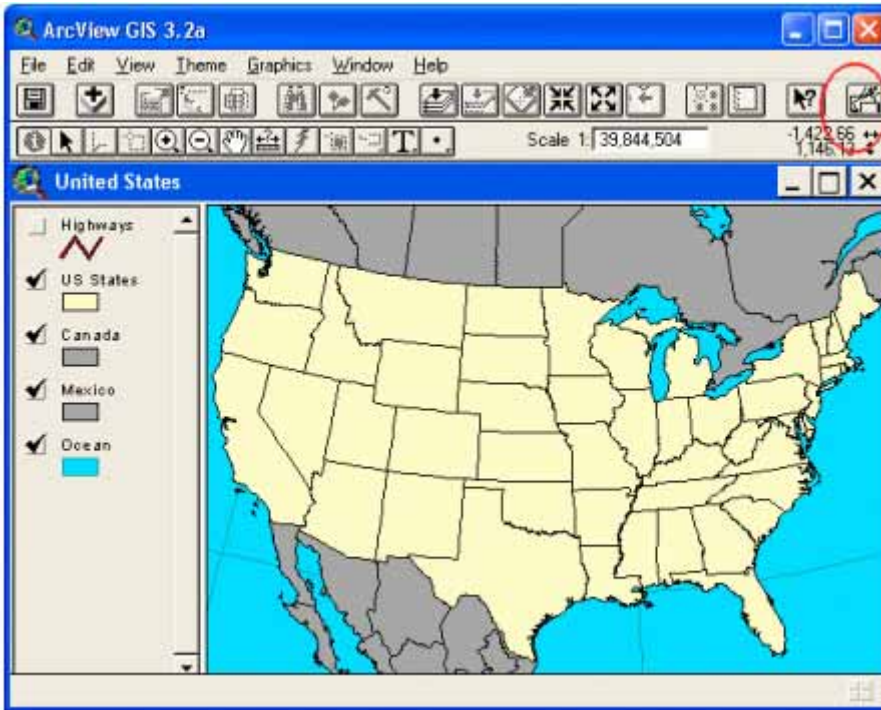
แล้วทำการปิดหน้าต่าง Customize: ออกไป

จะลองเริ่มทำการทดสอบชุดคำสั่ง GUI ดังนี้

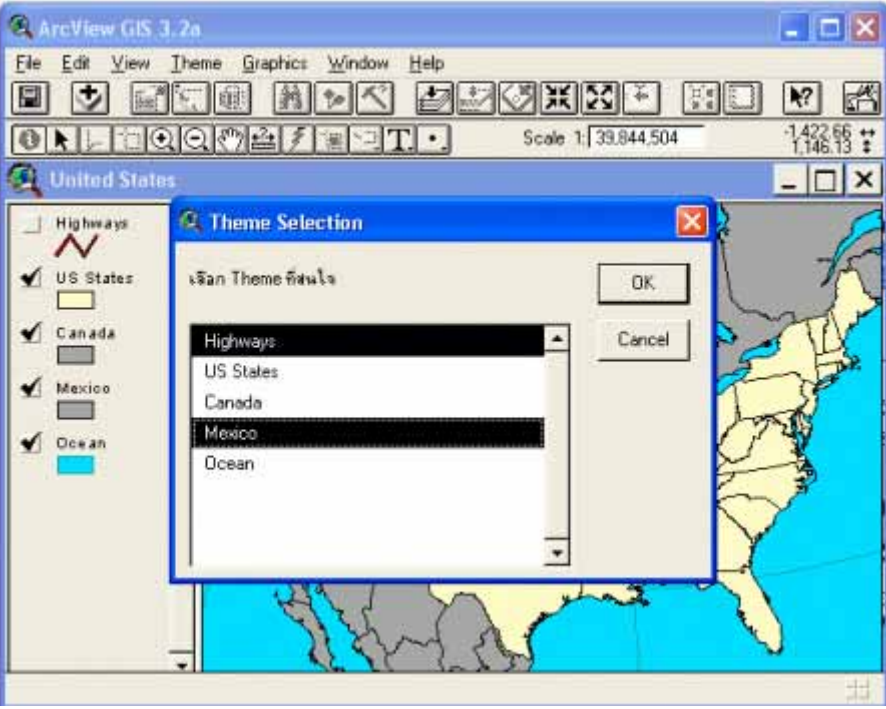
ให้ทำการเลือก Click ที่ปุ่ม A หรือ U หรือ W ปุ่มใดปุ่มหนึ่ง จะเป็นการเรียก View จากครั้งที่แล้วที่ได้เตรียมไว้ เช่นกดปุ่ม U



จะปรากฏหน้าต่างต่าง View ขึ้นมา และจะมีปุ่มที่เราสร้างไว้ใน View Window เมื่อสักครูให้กดปุ่มนั้น



ซึ่งจากรูปเราอาจจะเลือก เปิด Theme ชื่อ Highways ให้แสดงผล และปิด Mexico ให้ไม่แสดงผล แล้วกดปุ่ม OK



ก็จะได้ผลสุดท้ายดังรูป ให้ทดสอบโดยปิด View นี้ แล้วไปเปิด View อื่นๆ เพื่อทดสอบโปรแกรมได้ เมื่อทดสอบเสร็จให้ทำการบันทึกโปรเจคโดยกดเมนู File - Save Project ด้วยครับ เพื่อเก็บไว้ใช้ต่อไปในบทถัดไป

เราสามารถนำชุดคำสั่งนี้ไปดัดแปลงต่อไปในบทถัดไปซึ่งจะได้กล่าวถึงต่อไปครับ

หวังว่าบทความนี้จะช่วยให้คุณได้เข้าใจคำสั่ง Avenue มากขึ้นครับ