# คู่มือปฏิบัติการ: UI ใน VR ด้วย Unity และ Meta XR SDK (สัปดาห์ที่ 8)

## 1 การเตรียมและการตั้งค่า Canvas (Preparation and Canvas Setup)

1. **Import Assets** ให้ทำการ import Assets ที่จำเป็นสำหรับสัปดาห์นี้เข้ามาใน Project
   * จะมี Folder ชื่อ **HPBar**, **Intercom** และ **SFX**
2. **สร้าง Canvas** ใน Hierarchy ให้ทำการ Right-click และเลือก **UI > Canvas**
   * **Canvas** คือส่วนที่ใช้สำหรับแสดงผล UI โดยเฉพาะ ซึ่งจะอยู่ในโลกของ Screen Space หรือโลกของจอ (Screen) ไม่เกี่ยวข้องกับโลกของเกม (World) โดยตรง
3. **ตั้งค่า Canvas Scaler** เลือก **Canvas** ใน Hierarchy แล้วไปที่ Inspector เพื่อปรับ Component **Canvas Scaler**
   * เปลี่ยน Mode จาก Constant Pixel Size เป็น **Scale With Screen Size** เพื่อให้ UI ขยายตาม Screen Scale
   * ตั้งค่า **Reference Resolution** เป็น **1920 x 1080** (มาตรฐาน 1080p)
   * ตั้งค่า **Screen Match Mode** เป็น **Expand** (เพื่อให้ UI ขยายหรือย่อขนาดตามสัดส่วนของภาพ โดยไม่พยายามรักษาขนาดเดิมไว้)

## 2 การสร้างและทดลองใช้ UI Elements เบื้องต้น (Creating and Experimenting with Basic UI Elements)

### 2.1 Image Component

1. **สร้าง Image** ใน Hierarchy ให้ Right-click ที่ **Canvas** และเลือก **UI > Image**
2. **เปลี่ยน Sprite** ที่ Component Image ให้กดปุ่มจุด ด้านหลังช่อง **Source Image**
   * เลือกภาพที่ต้องการใช้ เช่น **Heart\_Icon.png**
   * สามารถเปลี่ยน **Color** หรือใส่ **Material** ได้ และสามารถกำหนดคุณสมบัติ **Raycast Target** (ถูกเมาส์จิ้มได้หรือไม่) สำหรับ Event ต่างๆ ได้
3. **ทำความเข้าใจ Anchor** **Anchor** คือจุดยึดที่กำหนดว่า UI Element นั้นๆ จะยึดตามส่วนใดของจอ (เช่น ซ้ายบน, ขวาบน, ตรงกลาง)
   * ลองปรับหน้าต่าง **Game** เป็น **Free Aspect** แล้วขยายจอ จะเห็นว่าถ้าไม่ตั้งค่า Anchor ภาพจะไม่ตามมุมซ้ายบน
4. **ตั้งค่า Anchor Preset** เลือก Image ใน Inspector ตรง **Anchor Preset** (ข้าง Rect Transform)
   * หากต้องการให้ภาพยึดตามมุมซ้ายบน (Top-Left) ให้กดเลือก **Top-Left**
   * หากต้องการให้ภาพยืดหรือหดตามขนาดของ Parent Component ให้เลือก **Stretch**
5. **ย้ายตำแหน่งพร้อมกำหนด Anchor/Pivot** หากต้องการย้ายตำแหน่งของภาพไปที่จุด Anchor พร้อมกัน เช่น ย้ายไปตรงกลางจอ ให้กดปุ่ม **Shift** และ **Alt/Option** ค้างไว้ แล้วกดเลือก **Anchor Preset** ที่ต้องการ (เช่น Center)
   * *ข้อควรระวัง:* Component Transform ของ UI คือ **Rect Transform** ซึ่งแตกต่างจาก Transform ทั่วไปของ Game Object
6. **การปรับขนาดด้วย Rect Tool** เลือก **Rect Tool (เลข 2)** ใน Toolbar
   * เครื่องมือนี้จะแสดงกรอบของ UI และหากปรับขนาด กรอบ (Width/Height) และ Position ของ Pivot ก็จะเปลี่ยนแปลงไป
   * หากต้องการย้าย Position ของ Pivot กลับมาตรงกลาง ให้กดปุ่ม **Shift** และ **Alt/Option** ค้างไว้ แล้วคลิกซ้าย

### 2.2 TextMeshPro Component

1. **สร้าง Text** Right-click ที่ Canvas เลือก **UI > Text - TextMeshPro**
   * หากมีการแจ้งเตือนให้ **Import** Asset ของ TextMeshPro ให้กด Import
   * เปลี่ยนข้อความเป็นคำว่า My Health
2. **ปรับข้อความและ Font Setting**
   * ปรับ **Font Asset** หากมีหลาย Font
   * ปรับ **Font Style** (Bold, Italic, Underline, Strikethrough)
   * ปรับ **Font Size**
   * เปิด **Auto Size** หากต้องการให้ Text ขยายขนาด Font เองตาม Rect Transform
   * ปรับ **Vertex Color** เพื่อใส่สี
   * ปรับ **Color Gradient** เพื่อทำสีแบบไล่เฉด (Horizontal/Vertical)
3. **ปรับ Layout**
   * ปรับ **Space Option** (ระยะห่างระหว่างข้อความ คำ แถว หรือ Paragraph)
   * ปรับ **Alignment** (ชิดซ้าย, ขวา, กลาง, บน, ล่าง)
   * ปรับ **Text Wrapping Mode** หากปรับเป็น **No Wrap** ข้อความจะไม่ขึ้นบรรทัดใหม่เมื่อเกินขอบ
   * ปรับ **Overflow** หากข้อความเกิน Rect Transform เช่น **Ellipsis** จะใส่จุด (...) ที่ท้ายข้อความ หรือ **Truncate** จะตัดข้อความทิ้งไปเลย
4. **ปรับ Material**
   * ใช้ **Outline** หรือ **Stroke** สำหรับเส้นรอบนอก
   * ใช้ **Underlay** หรือเงาด้านล่าง

### 2.3 ส่วนประกอบ UI อื่นๆ

* **Panel** เป็นแผ่นสำหรับปิดหรือจัดกลุ่ม UI Element
* **Toggle** เป็นปุ่มเปิด/ปิด (On/Off)
* **Slider** เป็นตัวสไลด์ที่ปรับค่าได้
  + สามารถเปิดดูส่วนประกอบด้านในได้ เช่น **Background**, **Fill Area** (ใช้สำหรับทำหลอดเลือด), และ **Handle Slide Area** (ปุ่มวงกลม)
* **Scroll View** เป็นแผงที่สามารถเลื่อนได้ (เช่น สำหรับแสดง Inventory ที่มีของเยอะๆ)
* **Button** เป็นปุ่มที่สามารถกดได้
  + สามารถกำหนด **Sprite** หรือ **Color** ได้
  + กำหนด **On Click** Event ซึ่งจะทำงานเมื่อมีการกดปุ่ม
* **Dropdown** เป็นเมนูแบบดึงลงมา
  + มีส่วนประกอบที่ปรับได้เช่น **Label** (ข้อความ), **Arrow** (ลูกศร), และ **Template** (รูปแบบของตัวเลือกเมื่อเปิด Dropdown)
* **Input Field** เป็นพื้นที่ที่สามารถพิมพ์ข้อความได้
  + สามารถกำหนด **Content Type** เพื่อจำกัดประเภทของข้อมูลที่ผู้ใช้สามารถป้อนได้ เช่น:
    - **Email Address**
    - **Password** (จะแสดงเป็นจุด)
    - **Integer Number** (ตัวเลขจำนวนเต็ม)
    - **Decimal Number** (ตัวเลขทศนิยม)

### 2.4 ทดสอบ Screen Space UI

1. กดปุ่ม **Play** เพื่อทดสอบ UI ที่สร้างในโหมด Screen Space
   * จะสามารถ Interact กับ Slider, Toggle, Button และ Input Field ได้ด้วย mouse และ keyboard

## 3 การเปลี่ยน UI เป็น World Space และการโต้ตอบใน VR (Switching to World Space and VR Interaction)

1. **เปลี่ยน Render Mode** เลือก **Canvas** ใน Inspector
   * ใน Component Canvas ตรง **Render Mode** ให้เปลี่ยนจาก Screen Space - Overlay เป็น **World Space**
2. **ปรับขนาดและตำแหน่ง**
   * ตั้งค่า Scale (X, Y, Z) เป็น **0.01** (ควรปลดล็อคเครื่องหมายหนีบออกก่อนเพื่อปรับค่าได้อิสระ)
   * ตั้งค่า Position เป็น **0, 0, 0** เพื่อให้อยู่ตรงกลางโลก
   * ทำการปรับ Position (ลากมาด้านหน้า) และ Scale (ย่อลง) เพื่อให้ UI อยู่ในขนาดที่เหมาะสมและสามารถ Interact ใน VR ได้
3. **เพิ่ม Ray Interaction** การโต้ตอบใน VR จะต้องเพิ่ม Component พิเศษเข้าไป
   * Right-click ที่ **Canvas** เลือก **Interaction SDK > Add Ray Interaction to Canvas**
   * เมื่อ **Setup Wizard** ปรากฏ ให้กด **Fix** (หากมีข้อความแจ้งเตือน PointableCanvasModule) และกด **Create**
4. **เพิ่ม Poke Interaction** การโต้ตอบด้วยการจิ้ม (นิ้วชี้)
   * Right-click ที่ **Canvas** เลือก **Interaction SDK > Add Poke Interaction to Canvas**
   * กด **Create**
   * ตอนนี้ Canvas จะสามารถถูก Interact ด้วย Ray-casting และ Poke (จิ้มด้วยมือ) ได้แล้ว
5. **ทดสอบการโต้ตอบ** กด Play แล้วใส่ Headset (VR)
   * ลองชี้ด้วย Controller (Ray) หรือจิ้มด้วยนิ้ว (Poke)
6. **ตั้งค่า System Keyboard สำหรับ Input Field**
   * Input Field ไม่สามารถทดสอบใน Unity Editor ได้ เพราะต้องใช้ **System Keyboard** (คีย์บอร์ดเสมือนของ Meta)
   * เลือก **Player (OVR Camera Rig)** ใน Hierarchy
   * ที่ Component **OVR Manager** ให้ติ๊กเลือก **Require System Keyboard** เมื่อ Build เกมออกไป คีย์บอร์ดจะแสดงขึ้นมาเมื่อกด Input Field
7. **กำหนดขอบเขต Canvas สำหรับ Interact (สำคัญ)**
   * การ Interact (Ray/Poke) จะทำงานตามขอบเขตขนาดของ **Canvas** ควรปรับขอบเขตให้พอดีกับ UI Elements จริงๆ เพื่อไม่ให้เกิดปัญหา
   * ใช้ **Rect Tool (เลข 2)** ใน Toolbar เพื่อปรับขอบเขตของ Canvas
   * *แก้ไข Anchor เมื่อปรับ Canvas:* หากการปรับ Canvas ทำให้ UI Elements ย้ายตำแหน่งเอง (เนื่องจาก Anchor พยายามยึดอยู่ตรงกลาง) ให้ทำการเปลี่ยน Anchor Preset ของ UI Elements ทั้งหมด (เช่น Image, Text) ให้เป็น **Top** หรือ **Center Top** โดยกด **Shift** ค้างไว้ขณะเลือก Anchor เพื่อเปลี่ยน Pivot ไปด้วย

## 4 การสร้าง Player Profile UI (HP Bar และ Score) (Player Profile UI Implementation)

### 4.1 การสร้าง HP Bar

1. **Duplicate Canvas** เลือก Canvas ที่ตั้งค่า World Space แล้ว กด **Ctrl + D** และปิด Canvas ตัวเดิมไว้
2. เปลี่ยนชื่อ Canvas ใหม่เป็น **Player Profile UI**
   * ซ่อน UI ทุกอย่างใน **Player Profile UI** เหลือแค่ Image และ Text
   * แก้ชื่อจาก Image เป็น Heart-Image
3. **สร้างโครงสร้าง HP Bar**
   * Right-click ที่ Player Profile UI เลือก **Create Empty** ตั้งชื่อว่า **HP Bar**
   * Right-click ที่ HP Bar เลือก **UI > Image** ตั้งชื่อว่า **Background**
   * Duplicate Background (Ctrl+D) ตั้งชื่อว่า **Fill**
4. **กำหนดภาพและ Fill**
   * เลือก **Fill** Image และกำหนด **Source Image** เป็น **Healthbar-Filled-Green.png**
   * เปลี่ยน **Image Type** จาก Simple เป็น **Filled**
   * กำหนด **Fill Method** เป็น **Horizontal** (สามารถลองใช้ Radial, Vertical, หรือ Radial 180/360 ได้ตามความเหมาะสม)
   * เลือก **Background** Image และกำหนด **Source Image** เป็น **Healthbar-Empty-Green.png**
5. **ปรับ Anchor ของภาพ**
   * เลือกทั้ง **Background** และ **Fill**
   * ตั้งค่า **Anchor Preset** เป็น **Stretch** (พร้อมกด Shift + Alt/Option ค้างไว้ เพื่อให้ภาพเต็ม Parent Container)
6. จัดวางองค์ประกอบ My Health Text, Heart Image และ HP Bar ให้สวยงามบน Canvas
   * ปรับ Anchor ของ UI Elements (เช่น HP Bar) ให้เหมาะสม เช่น **Top** หรือ **Bottom**

### 4.2 Scripting สำหรับ HP Bar

1. **สร้าง Script** ใน Folder Script ให้ Create C# Script ชื่อ **HPBar**
2. **แก้ไข Script HPBar**
   * เพิ่ม Library: using UnityEngine.UI;
   * ประกาศตัวแปร:
   * public Image hpFill;
   * public Health health;
   * *หมายเหตุ:* ต้องตรวจสอบ Script Health ที่มีอยู่แล้ว และเปลี่ยนตัวแปร **maxHealth** และ **currentHealth** ให้เป็น public เพื่อให้ Class อื่นสามารถดึงค่าไปใช้งานได้
   * กลับไปที่ไฟล์ **HPBar.cs** ในฟังก์ชัน **Update** ให้กำหนดค่า fillAmount:
   * hpFill.fillAmount = health.currentHealth / health.maxHealth;

(เป็นการนำ Current Health หาร Max Health เพื่อหาค่าเปอร์เซ็นต์ (0-1) สำหรับ Fill Amount)

1. **เชื่อมต่อ Script**
   * ลาก Script **HPBar** ใส่ที่ GameObject **HPBar**
   * ลาก Image **Fill** ใส่ในช่อง **Hp Fill**
   * ลาก GameObject **Player** ใส่ในช่อง **Health** (เนื่องจาก Player มี Component Health ติดอยู่แล้ว)
2. **ทดสอบ** กด Play แล้วลองเปิดประตูให้ Enemy โจมตี แล้วกลับเข้ามาหลังประตู จะเห็นหลอดเลือดลดแบบ Real-time

### 4.3 การติด UI เข้ากับ Hand (Hand Attached UI)

1. **เพิ่ม Component Hand Joint** เลือก **Player Profile UI**
   * Add Component ชื่อ **Hand Joint**
2. **กำหนด Hand และ Joint**
   * ในช่อง **Hand** ให้ลาก **LeftInteractions** จาก Player ใส่เข้าไป
   * ในช่อง **Joint ID** ให้เลือกจุดที่ต้องการให้ UI ยึดติด เช่น **Hand Wrist Root** (ข้อมือ)
3. **ปรับ Offset ขณะ Play Mode**
   * กด Play
   * ปรับ **Scale** ของ Player Profile UI ให้เหมาะสมกับขนาดมือ
   * ปรับ **Pos Offset** และ **Rot Offset** บน Component **Hand Joint** จนกระทั่ง UI อยู่ในตำแหน่งที่ต้องการ (เช่น อยู่บนมือแทนที่จะอยู่บนข้อมือ)
4. **บันทึกค่า Offset (สำคัญ)**
   * ขณะที่ยังอยู่ใน Play Mode ให้คลิกที่ 3 จุด (Three-dot menu) ที่ Component **Hand Joint** เลือก **Copy Component**
   * หยุด Play Mode
   * คลิกที่ 3 จุดเดิม เลือก **Paste Component Values** ค่า Offset ที่ปรับไว้จะถูกบันทึก

### 4.4 การสร้าง Score Text

1. **สร้าง Score Text** Right-click ที่ Player Profile UI เลือก **UI > Text - TextMeshPro** ตั้งชื่อว่า **Score-Text**
   * กำหนดข้อความเบื้องต้น เช่น "Score: 0"
2. **สร้าง Script Score Manager** Create C# Script ชื่อ **ScoreManager**
   * ประกาศตัวแปร: public int currentScore;
   * สร้างฟังก์ชันสำหรับเพิ่มคะแนน:
   * public void AddScore(int scoreAmount)
   * {
   * currentScore += scoreAmount;
   * }
3. **สร้าง GameObject** Create Empty GameObject ชื่อ **ScoreManager** ใน Hierarchy และลาก Script **ScoreManager** ใส่เข้าไป
4. **เชื่อมต่อ Enemy Death Event**
   * เลือก Mutant\_Model (Enemy) GameObject
   * ที่ Event **On Dead** กดเครื่องหมายบวก (+)
   * ลาก **ScoreManager** GameObject ใส่ในช่อง Object
   * เลือก Function **ScoreManager > AddScore(int)** และใส่ค่าคะแนนที่ต้องการ (เช่น 100)
5. **สร้าง Script Score Text** Create C# Script ชื่อ **ScoreText**
   * เพิ่ม Library using TMPro; (หากจำเป็น)
   * ประกาศตัวแปร:
   * public TextMeshProUGUI scoreText;
   * public ScoreManager scoreManager;
   * ในฟังก์ชัน **Update** ให้ทำการ Update ข้อความ:
   * scoreText.text = "Score: " + scoreManager.currentScore;
6. **เชื่อมต่อ Script ScoreText**
   * ลาก Script **ScoreText** ใส่ที่ GameObject **Score-Text**
   * ลาก GameObject **Score-Text** ใส่ในช่อง **Score Text**
   * ลาก **ScoreManager** GameObject ใส่ในช่อง **Score Manager**
7. **ทดสอบ** กด Play และยิง Enemy ให้ตาย คะแนนจะเพิ่มขึ้น

## 5 การสร้างปุ่ม Restart (Restart Button Implementation)

1. **สร้างปุ่ม** Right-click ที่ Player Profile UI เลือก **UI > Button - TextMeshPro**
   * ตั้งชื่อ GameObject ว่า **Restart-Button**
   * แก้ไข Text ให้แสดงคำว่า **Restart**
2. **สร้าง Script Scene Loader** Create C# Script ชื่อ **SceneLoader**
3. **สร้าง GameObject** Create Empty GameObject ชื่อ **SceneLoader** ใน Hierarchy และลาก Script **SceneLoader** ใส่เข้าไป
4. **แก้ไข Script Scene Loader**
   * เพิ่ม Library: using UnityEngine.SceneManagement;
   * สร้างฟังก์ชันสำหรับ Restart Scene ปัจจุบัน:
   * public void RestartCurrentScene()
   * {
   * string currentSceneName = SceneManager.GetActiveScene().name;
   * SceneManager.LoadScene(currentSceneName);
   * }
5. **ตั้งค่า Build Settings (สำคัญ)**
   * ไปที่ **File > Build Profiles**
   * เปิด **Scenes List** List และกด **Add Open Scenes** เพื่อเพิ่ม Scene ปัจจุบันเข้าไปใน Index (Scene แรกจะเป็น Index 0 เสมอ)
6. **เชื่อมต่อ Button On Click**
   * เลือก **Restart-Button** GameObject
   * ใน Event **On Click**() ลาก **SceneLoader** GameObject ใส่เข้าไป
   * เลือก Function **SceneLoader > RestartCurrentScene**
7. **ปรับขอบเขต Canvas** ใช้ **Rect Tool** เพื่อขยายขอบเขตของ **PlayerProfileUI** Canvas ให้ครอบคลุม Restart Button เพื่อให้สามารถ Interact ได้
8. **ทดสอบ** กด Play และลองกดปุ่ม Restart Scene

## 6 การใช้งาน Intercom (Intercom Implementation)

1. **ลาก Intercom Prefab** ลาก **Intercom prefab** จาก Assets Week 8 เข้ามาใน Scene
   * ลาก Intercom prefab ใน scene ไปอยู่ใกล้ ๆ **Intercom\_01** (ตัวที่เราใช้การ์ดไปทาบเพื่อเปิดประตู)
   * ทำการปิด GameObject **Intercom\_01** ก่อน (Set Active: false)
2. **ตั้งค่ารหัสผ่าน** เลือก Component Intercom (Script) (บน Intercom Prefab)
   * กำหนด **Correct Password** (รหัสผ่านที่ถูกต้อง) เช่น "1111”
3. **กำหนด Event เมื่อรหัสถูกต้อง**
   * ที่ Event **On Submit Correct Password** กด **+** และลาก **Intercom\_01** GameObject ใส่เข้าไป
   * เลือก Function **GameObject > Set Active (bool)** และตั้งค่าเป็น **true** (เพื่อเปิด Intercom เมื่อใส่รหัสถูก)
4. **กำหนด Event เมื่อรหัสผิด**
   * ที่ Event **On Submit Incorrect Password** กด **+** และลาก **Intercom\_01** GameObject ใส่เข้าไป
   * เลือก Function **GameObject > Set Active (bool)** และตั้งค่าเป็น **false** (เพื่อปิด Intercom เมื่อใส่รหัสผิด)
5. **ทดสอบ** รัน Play Mode แล้วลองใช้ Poke หรือ Ray กดปุ่มบน NumPad และกด Submit

* *คำอธิบายเพิ่มเติมเกี่ยวกับ Script Intercom:*
  + Script **Intercom.cs** จะ Reference ถึง Input Field, Reset Button และ Submit Button
  + โดยจะใช้คำสั่ง AddListener เพื่อผูกฟังก์ชัน ResetPassword (เคลียร์ Text ใน Input Field) และ SubmitPassword (เช็ครหัสที่ป้อนเทียบกับ Correct Password) เข้ากับ Event On Click ของปุ่ม Reset และ Submit
  + Script **NumpadButton.cs** จะผูกฟังก์ชันเพื่อเพิ่มค่า (padCharacter) เข้าไปใน Input Field ของ Intercom เมื่อมีการกดปุ่มตัวเลข (0-9)