Interaction 2: Implementation

Notes for the SIT-DP module: **Developing Immersive Applications**

Created by: Chek Tien TAN

slido

Please download and install the
Slido app on all computers you use

Join at slido.com
#3446130

ⓘ Start presenting to display the joining instructions on this slide.

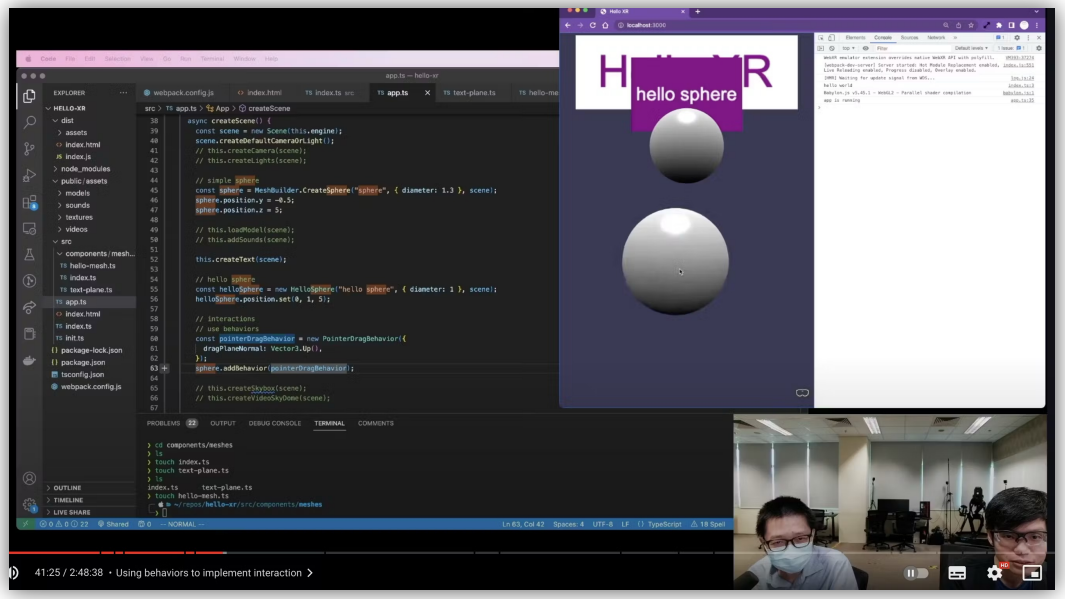
Learning Objectives

- Differentiate code constructs (behaviours, actions and observables) to implement interactions in WebXR
- Implement various typical object handling interactions in WebXR
- implement various typical locomotion interactions in WebXR

Behaviors

https://doc.babylonjs.com/features/featuresDeepDive/behaviors/

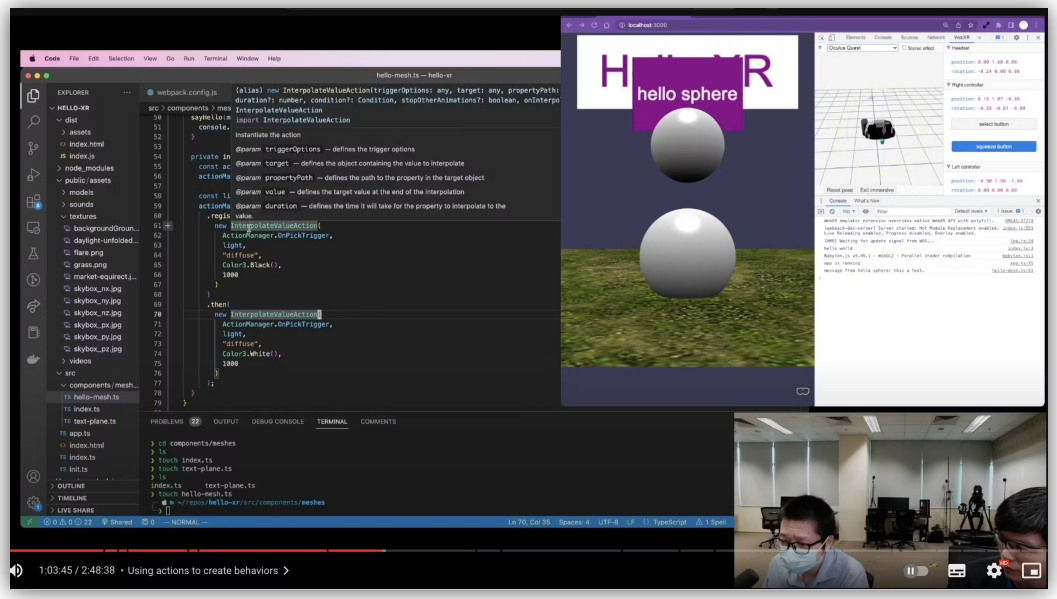
- Predefined, reusable interactions without custom code
- Common interactions like dragging, scaling, following, etc.



ActionManager

https://doc.babylonjs.com/features/featuresDeepDive/events/actions

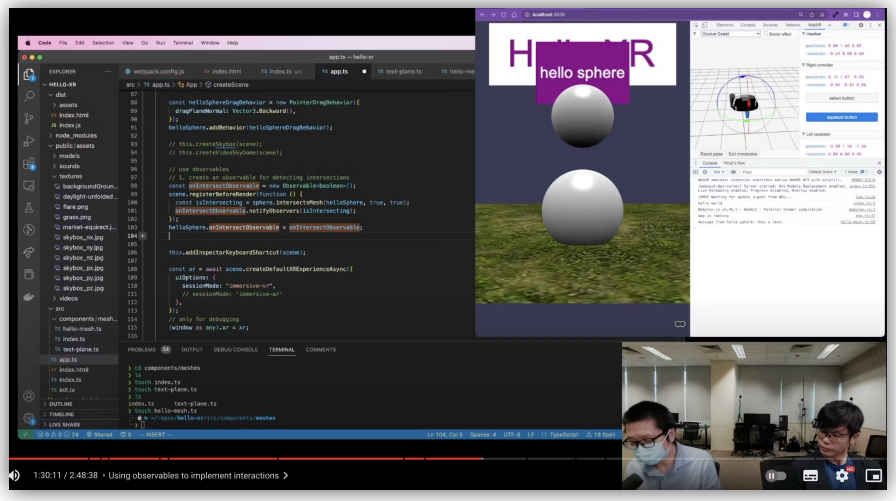
- Define property changes triggered by pre-defined events
- Customize interaction parameters (e.g., duration, conditions, triggers)



Observables

https://doc.babylonjs.com/features/featuresDeepDive/events/observables/

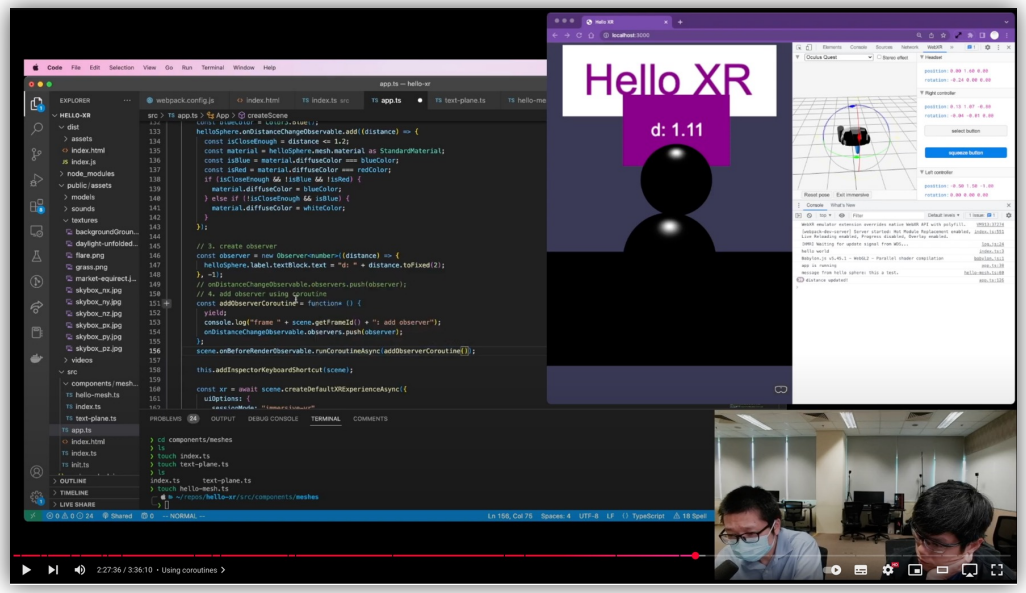
- General code construct for observer pattern
- Subscribe and receive notifications to events
- Fully customizable interactions



Coroutines

https://doc.babylonjs.com/features/featuresDeepDive/events/coroutines/

- Modern “concurrency” through generators: `function*()`
- yield to pause/resume execution per frame

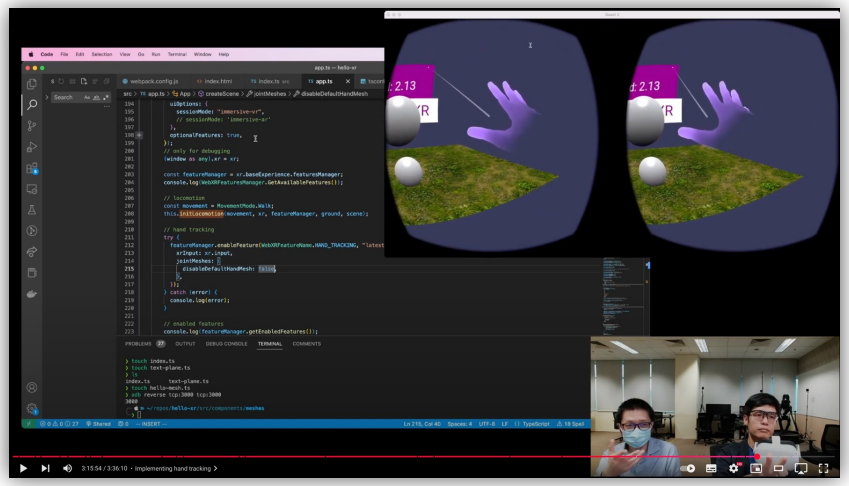


Common XR Interactions

https://doc.babylonjs.com/typedoc/classes/BABYLON.WebXRFeaturesManager

https://doc.babylonjs.com/features/featuresDeepDive/mesh/gizmo

- WebXRFeaturesManager: pre-made XR features, e.g.,
  - WebXRTeleportation
  - WebXRHandTracking, etc.
- GizmoManager: common 3D UI editing tools



slido

Please download and install the Slido app on all computers you use

You want to create a button in your Babylon.js scene that, when touched, makes a door open with a creaking sound that lasts 0.5 seconds.

Which implementation approach is the most straightforward without reinventing the wheel?

ⓘ Start presenting to display the poll results on this slide.

You want to create a button in your Babylon.js scene that, when touched, makes a door open with a creaking sound that lasts 0.5 seconds.

Which implementation approach is the most straightforward without reinventing the wheel?

- Behaviors

  0%

- ActionManager

  0%

- Observables

  0%

slido

Please download and install the Slido app on all computers you use

Implement a jump action in your Babylon.js scene when the user presses the keyboard spacebar. Which trigger should you use in the ActionManager?

ⓘ Start presenting to display the poll results on this slide.

Implement a jump action in your Babylon.js scene when the user presses the keyboard spacebar. Which trigger should you use in the ActionManager?

- OnPickTrigger

  0%

- OnIntersectionEnterTrigger

  0%

- OnKeyUpTrigger

  0%

- NothingTrigger

  0%

slido

Please download and install the Slido app on all computers you use

In your Babylon.js scene, you need to periodically track changes in the position of a dog object and automatically show updates on the HUD based on its proximity to different objects.

Which implementation approach is the most straightforward without reinventing the wheel?

ⓘ Start presenting to display the poll results on this slide.

In your Babylon.js scene, you need to periodically track changes in the position of a dog object and automatically show updates on the HUD based on it's proximity to different objects.

Which implementation approach is the most straightforward without reinventing the wheel?

- Behaviors

  0%

- ActionManager

  0%

- Observables

  0%

slido

Please download and install the Slido app on all computers you use

In your Babylon.js scene, when a pen mesh and a paper mesh touch each other (i.e., intersect), you want to show virtual ink appearing.

Which implementation approach is the most straightforward without reinventing the wheel?

ⓘ Start presenting to display the poll results on this slide.

In your Babylon.js scene, when a pen mesh and a paper mesh touch each other (i.e., intersect), you want to show virtual ink appearing.

Which implementation approach is the most straightforward without reinventing the wheel?

- Behaviors

  0%

- ActionManager

  0%

- Observables

  0%

slido

Please download and install the Slido app on all computers you use

In total, how many observers were used here?

```
const onDistanceChangeObservable = new Observable<number>();

let previousDistance: number;
scene.onBeforeRenderObservable.add(() => {
    const currentDistance = Vector3.Distance( sphere.position, Vector3.Zero());

    if (currentDistance !== previousDistance) {
        previousDistance = currentDistance;
        onDistanceChangeObservable.notifyObservers(currentDistance);
    }
});

onDistanceChangeObservable.add(distance => {
    helloText.text = `d: ${distance.toFixed(2)}`;
});
```

ⓘ Start presenting to display the poll results on this slide.

In total, how many observers were used here?

Show Image | Show poll options

0 — 0%

1 — 0%

2 — 0%

3 — 0%

4 — 0%

5 — 0%

slido

Please download and install the Slido app on all computers you use

In total, how many observables did we operate on?

```
const onDistanceChangeObservable = new Observable<number>();

let previousDistance: number;
scene.onBeforeRenderObservable.add(() => {
    const currentDistance = Vector3.Distance( sphere.position, Vector3.Zero());

    if (currentDistance !== previousDistance) {
        previousDistance = currentDistance;
        onDistanceChangeObservable.notifyObservers(currentDistance);
    }
});

onDistanceChangeObservable.add(distance => {
    helloText.text = `d: ${distance.toFixed(2)}`;
});
```

ⓘ Start presenting to display the poll results on this slide.

In total, how many observables did we operate on?

Show Image | Show poll options

0 — 0%

1 — 0%

2 — 0%

3 — 0%

4 — 0%

5 — 0%

slido

Please download and install the Slido app on all computers you use

In total, how many observables did we create?

```
const onDistanceChangeObservable = new Observable<number>();

let previousDistance: number;
scene.onBeforeRenderObservable.add(() => {
    const currentDistance = Vector3.Distance( sphere.position, Vector3.Zero());

    if (currentDistance !== previousDistance) {
        previousDistance = currentDistance;
        onDistanceChangeObservable.notifyObservers(currentDistance);
    }
});

onDistanceChangeObservable.add(distance => {
    helloText.text = `d: ${distance.toFixed(2)}`;
});
```

ⓘ Start presenting to display the poll results on this slide.

In total, how many observables did we create?

- 0 — 0%
- 1 — 0%
- 2 — 0%
- 3 — 0%
- 4 — 0%
- 5 — 0%

slido

Please download and install the Slido app on all computers you use

What is the mechanics of the following code?

```
pointerDragBehavior.onDragObservable.add(eventData => {
    console.log(sphere.position);
});
```

ⓘ Start presenting to display the poll results on this slide.

What is the mechanics of the following code?

- It adds an Observable to pointerDragBehavior of the sphere

  0%

- It adds an Observer to the sphere

  0%

- It adds an Observer to the onDragObservable of the pointerDragBehavior

  0%

- It adds an Observable to the sphere

  0%

slido

Please download and install the Slido app on all computers you use

What is the order of the console logs in the following Babylon.js code?
(Assume the rest of the code is correct and the scene is set up properly)

```
const outOfOrder = function* () {
    (async function () {
        await Tools.DelayAsync(3000);
        console.log('1');
    })();
    yield;
    (async function () {
        console.log('2');
    })();
    yield;
    (async function () {
        await Tools.DelayAsync(1000);
        console.log('3');
    })();
    yield;
    (async function () {
        await Tools.DelayAsync(2000);
        console.log('4');
    })();
};
scene.onBeforeRenderObservable.runCoroutineAsync(outOfOrder());
```

ⓘ Start presenting to display the poll results on this slide.

What is the order of the console logs in the following Babylon.js code?
(Assume the rest of the code is correct and the scene is set up properly)

- 1, 2, 3, 4 — 0%
- 2, 3, 4, 1 — 0%
- 1, 4, 3, 2 — 0%
- 4, 3, 2, 1 — 0%

slido

Please download and install the Slido app on all computers you use

Which API class in Babylon.js will allow you to easily add UI controls to easily manipulate the position, rotation, and scale of meshes in your scene?

ⓘ Start presenting to display the poll results on this slide.

Which API class in Babylon.js will allow you to easily add UI controls to easily manipulate the position, rotation, and scale of meshes in your scene?

- MultiPointerScaleBehavior

  0%

- GizmoManager

  0%

- PointerDragBehavior

  0%

- WebXRFeaturesManager

  0%

slido

Please download and install the Slido app on all computers you use

What does timeToTeleport do in the following Babylon.js code?

ⓘ Start presenting to display the poll results on this slide.

What does timeToTeleport do in the following Babylon.js code?

- sets the duration of the teleportation animation

  0%

- sets the maximum time to complete the teleportation

  0%

- sets the minimum delay between each teleportation trigger

  0%

- sets the time in to hold the button before teleportation triggers

  0%

slido

Please download and install the
Slido app on all computers you use

Audience Q&A

ⓘ Start presenting to display the audience questions on this slide.