像素惡地進階使用手冊

Pixel Badlands Advanced User Manual

Introduction to gather.town

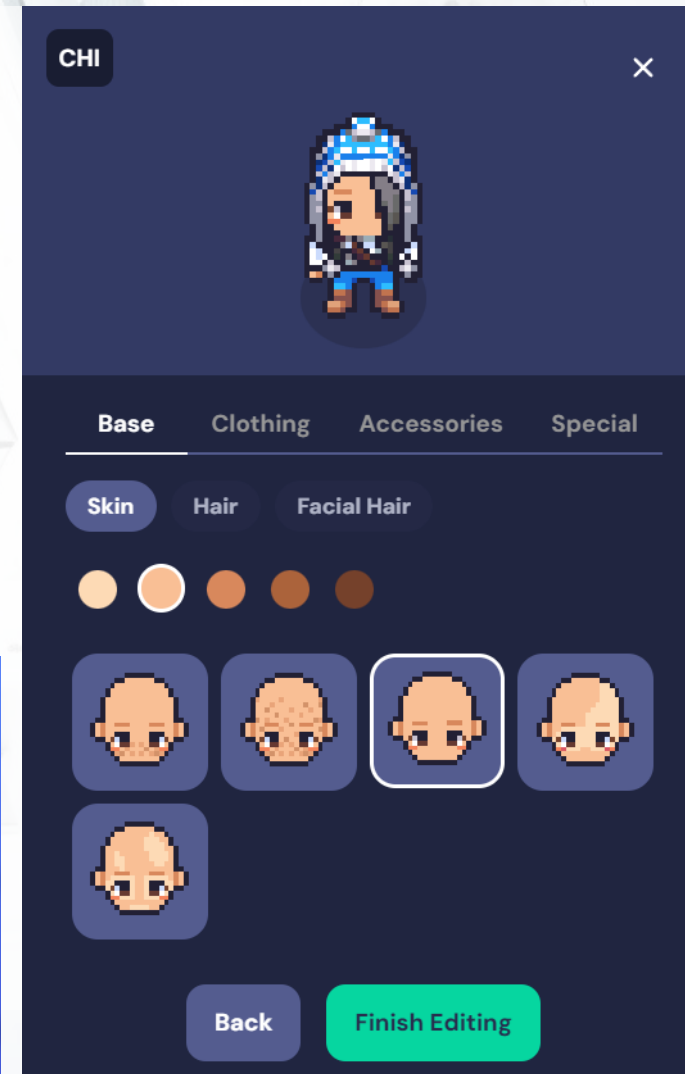Prof. Dr. & Dipl. Des. Wei-Chi Chien

Lab of Humanistic Experience and Technology Design| NCKU

More than just Gather.Town

- ⊞ basic concepts about metaverse and web3 technology

- ⊞ tele-presence and gamified social interaction

- ⊞ prototyping metaverse

- ⊞ high-level object-oriented programming (JavaScript and TypeScript)

- ⊞ potential of an experience-oriented and sustainable innovation

Your First Impression about Gather.Town

- create your avatar

- use arrow keys to move

- use x key to interact with an object

- stay close to other player to talk

- play with f, g, z, and number 1-7 keys

Your First Impression about Gather.Town

- Formosa
  https://app.gather.town/app/SAzp2TxhnEakR34j/Formosa

- Picture Book
  https://app.gather.town/app/YeDXPzJe54TvkIUr/Picture%20Book

- Education Explorers
  https://app.gather.town/app/IaXahXk5toVPwFbs/explorers

- Sternburg Museum
  https://app.gather.town/app/LVoLpXHUvT5P8LAA/Sternberg%20Museum

- Folkwang UdK
  https://app.gather.town/app/7rmncLDnXyh2ZFc6/salden-folkwang

- NCKU library
  https://app.gather.town/app/UYjLS4B93uOX4l8n/online%20exhibition

像素惡地進階使用手冊

Pixel Badlands Advanced User Manual

Console & Syntax

Prof. Dr. & Dipl. Des. Wei-Chi Chien

Lab of Humanistic Experience and Technology Design| NCKU

To Our Coding Practice in this Course

⊞ *programming language: JavaScript, nodeJS, TypeScript*

learning goal:

⊞ *use online reference to program your interactive mechanism*

⊞ *use browser console to run your code and interact with the **API***

⊞ *with fundamental know, learn coding on your own*

⊞ *properly express your technical problems*

E	SpriteDirectionEnum_ENUM
E	TileType
E	TriviaGamePhase
E	VehicleAction
C	Engine
C	Game
C	Player
C	PlayerDBFields
C	SlidingWindow
C	WallFloorDB
I	AccessRequest
I	AccessRequestRespondedTo

```
$ file $(which python3)
/opt/homebrew/bin/python3: Mach-O 64-bit executable arm64
```

Node canvas dependencies (M1 only)

node-canvas does not have a prebuilt binary for the M1 architecture and will attempt to build from source. Additional dependencies will be required to compile, install through homebrew.

```
$ brew install pkg-config cairo pango libpng jpeg giflib librsvg
pixman
```

Node

Make sure you're at node version 16.20.0 exactly.
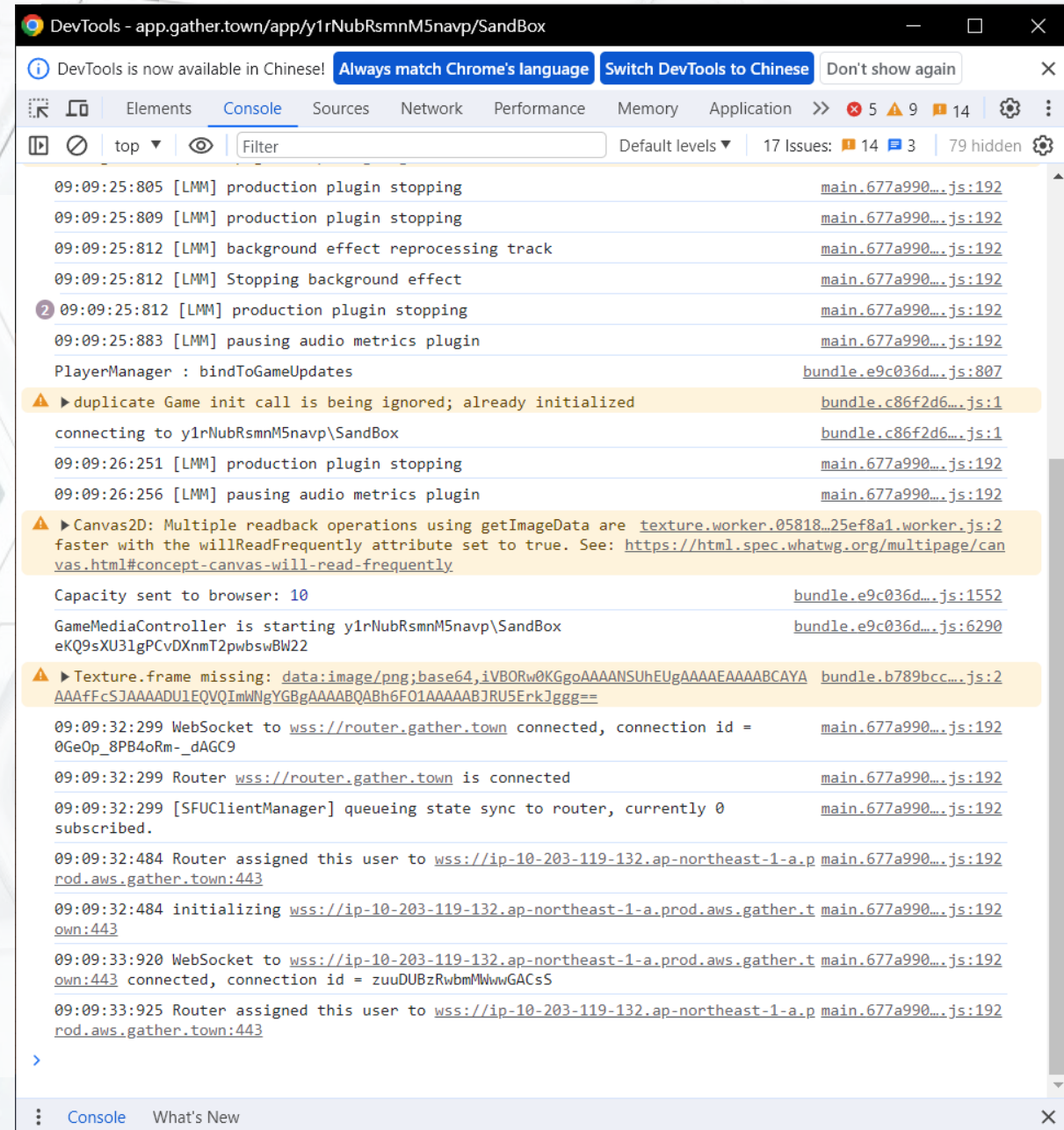
NVM can be installed by following the steps at https://github.com/nvm-sh/nvm#installing-and-

Console

⊞ *Chrome: F12*

⊞ *Edge: Ctrl+Shift + i*

⊞ *Firefox: Ctrl+Shift + j*
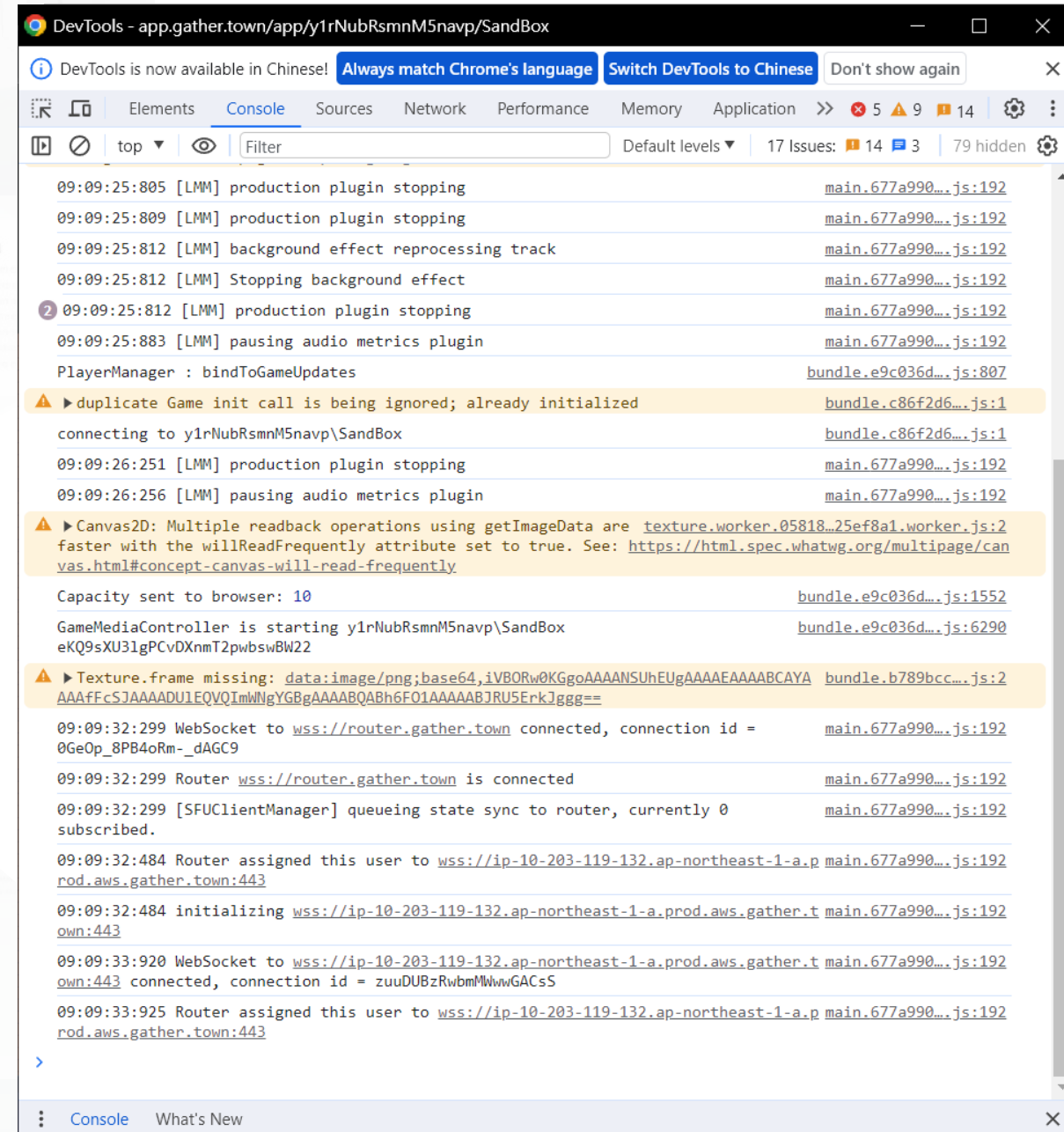
⊞ *Safari: Cmd+Opt + c*

Console: Practice

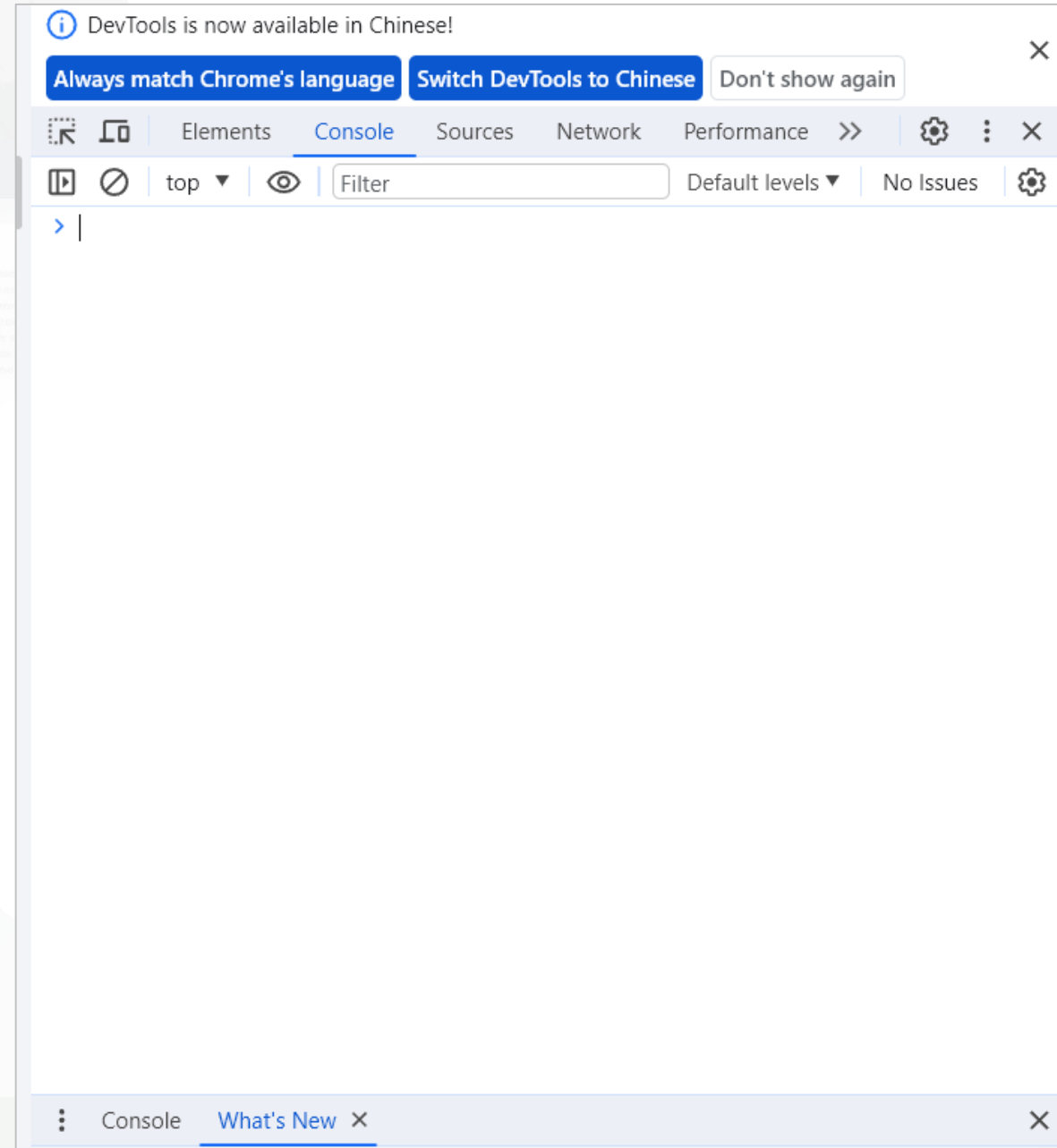⊞ *Open your console in browser.*

⊞ *Go to your Gather.Town space.*

Console: Practice

Try the following command lines.

➤ 1 + 1

➤ Hello!

➤ console.log("Hello!");

➤ *Console is an interface to interact with the internet service.*

Console: Practice

Try the following command lines.

➢ let myVariable = 7;

➢ console.log(myVariable);

➢ console.log('myVariable');

➢ *Note the difference between the variables and strings.*
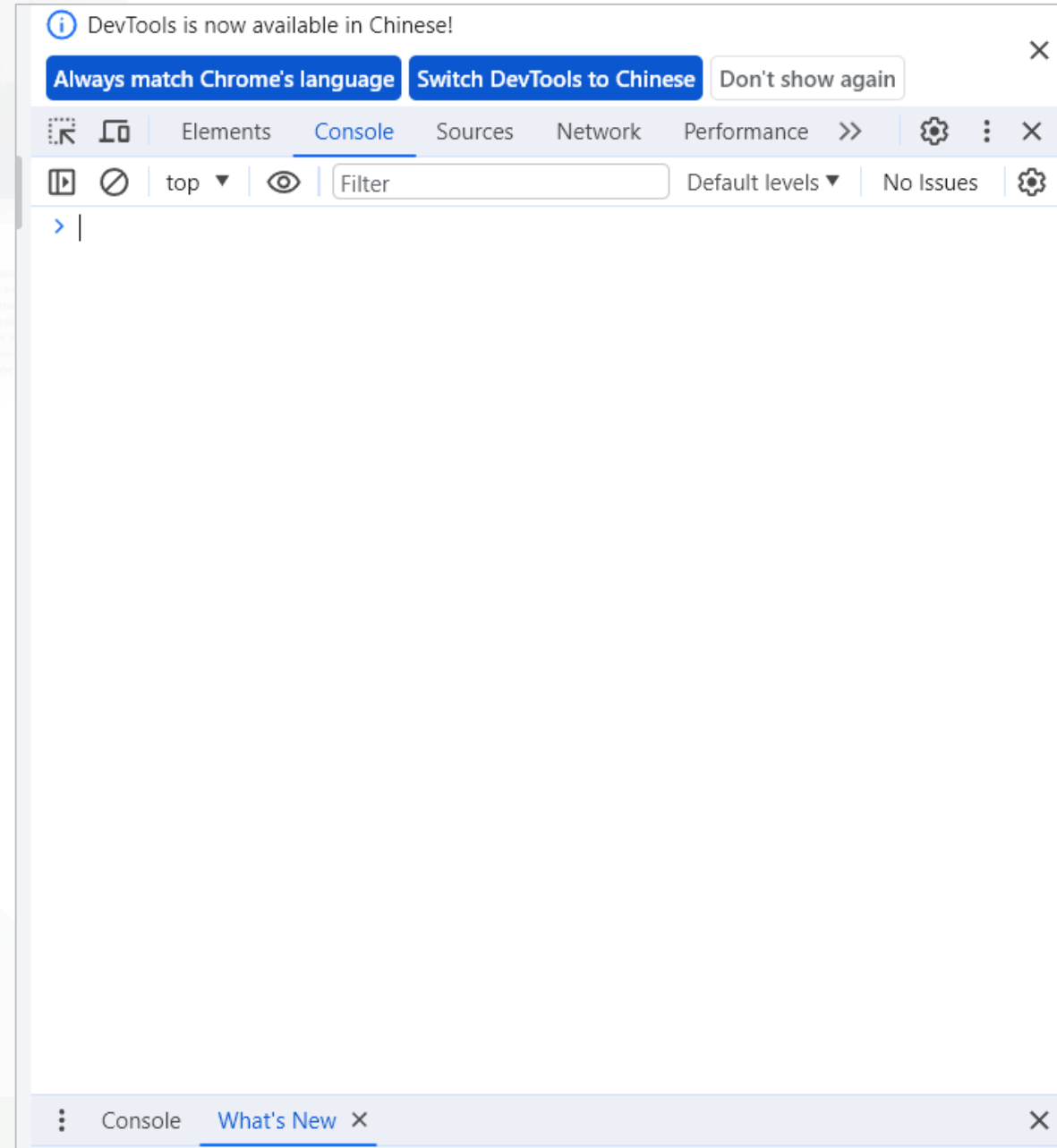
Gather.Town API terminology

Space

Room/Map
- Item
- Item
- Item

Player

Player

Room/Map
- Item
- Item

Player

Player   Player

Room/Map
- Item
- Item
- Item
- Item

Player
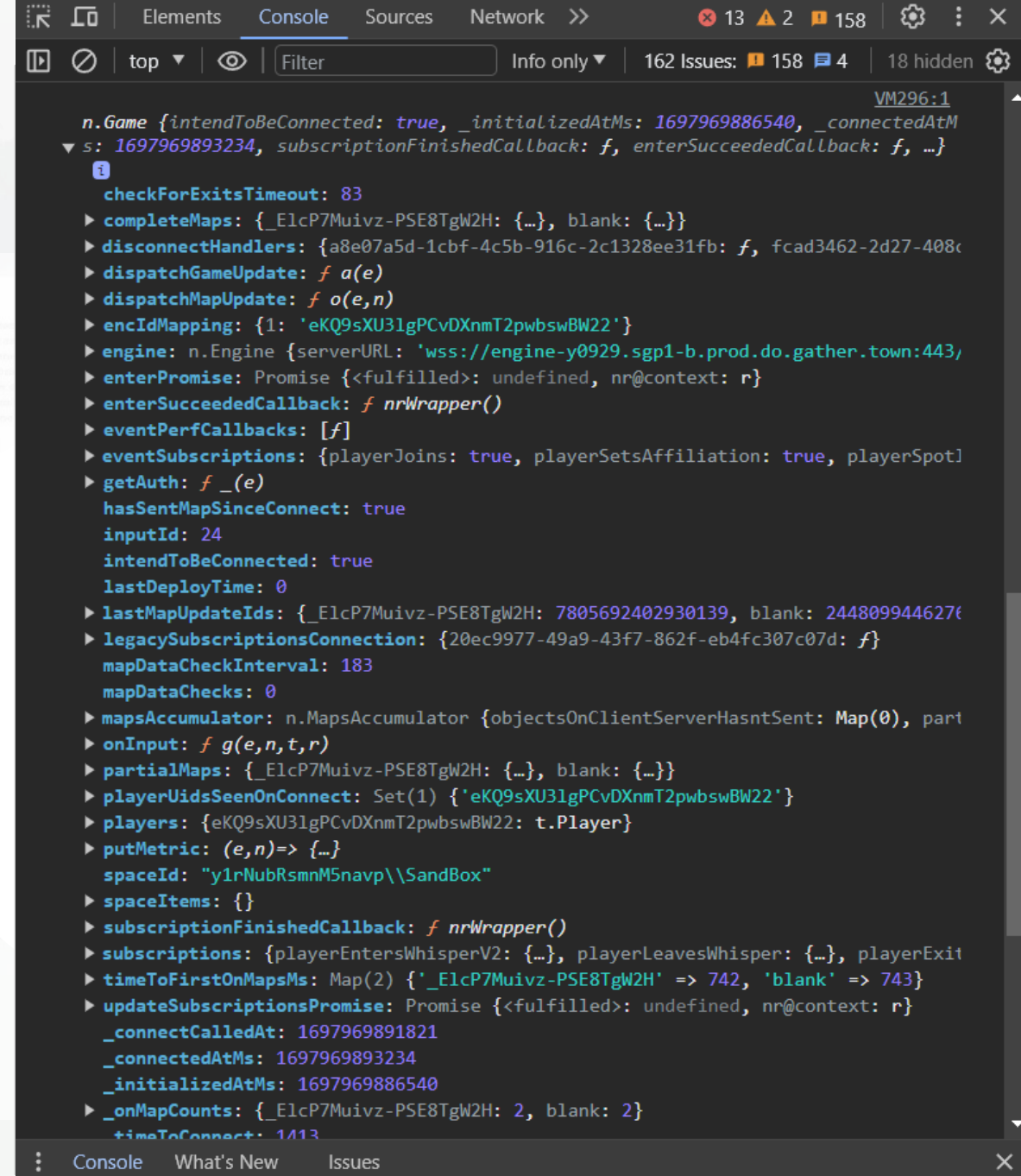
Player

Gather.Town API terminology

- ⊕ *functions*
- ⊕ *parameters*
- ⊕ *properties*
- ⊕ *objects*
- ⊕ *methods*
- ⊕ *class*

Console: Practice

Try the following command lines.

➢ console.log(game);

➢ game;

➢ console.log(game.players);

➢ console.log(game.**P**layers);

What is Syntax?

(human language)

buy some milk.

(computer language)

buy (some milk);

function

parameter

// statement, command

⊞ **Syntax is the gramma of computer's language to talk.**

What is Syntax?

(human language)

The mother is buying some milk.

(computer language)

mother.buy (some_milk);

object

parameter

function/method

What is Syntax?

(human language)

The mother is buying some milk.

(computer language)

`mother.buy (some milk, supermarket);`

object

function/method

parameter 1

parameter 2

What is Syntax?

(human language)

a bottle of milk, 300 mL, NT. 200, made in Taiwan, 10/8 expiry date

(computer language)

object

```
milk = {
        unit: 'bottle',
        number: 1,
        amount: 300,
        price: 200,
        production location: 'Taiwan',
        expiry date: '10/8'
};
```

value as String

value as number

property

What is Syntax?

(human language)

a bottle of milk, 300 mL, NT. 200, made in Taiwan, 10/8 expiry date

(computer language)

```
milk = {
        unit: 'bottle',
        number: 1,
        amount: 300,
        price: 200,
        production location: 'Taiwan',
        expiry date: '10/8'
};
```

Number

String

What is Syntax?

(human language)

my mother is 40 years old, a housewife, often asks me to buy something for her, and do the laundry for us...

(computer language)

```
mother = {
        age: 40,
        career: 'housewife,
        buy: function(something, from somewhere){
                … …;
        }
        doLaundry: function(…){
                … …;
        }
};
```

properties →

methods/ function →
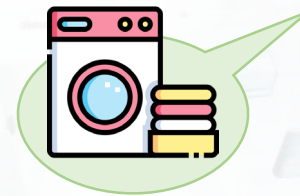
What is Syntax?

⊕ *The hero is fighting against the dragon. The dragon is breathing fire towards the hero.*

```
hero.fight(dragon);

dragon.fireBreathing(hero);
```

object

methods/
function

parameter

What is Syntax?

⊞ *The hero is unmarried, 19 years old, and can fight with sword or defend with shield.*

```
hero = {
        marriage: false;
        age: 19;
        swordFight: function(…){
                …
        }
        shieldDefend: function(…){
                …
        }
};
```

object

property

methods/ function

Gather.Town API terminology

In programming language...

⊕ *FUNCTION makes things happen. It performs under specific condition and may be oriented to the situations.*

⊕ *PARAMETERs are the required reference information for a function. Sometimes, a function do not require any parameter.*

⊕ *OBJECT is a collection of functions and properties.*

⊕ *METHOD is the function packaged in an object.*

⊕ *To use the method of an object, we need to **CALL** the object and then the method.*

gather-game-client

Coding and Syntax

console.log(game.players**)**

➢ *There is a object called "console" and a method (function) called "log".*

➢ *This method is used to display some information.*

➢ *There is one parameter in this method, which is the information to be displayed.*

Coding and Syntax

*console.log(**game.players**)*

➢ game *is an object.*

➢ players *is a property in the game object that stores the information of players.*

➢ *A significant difference between function and property is that function is followed by brackets:* console.log() *, while property or object do not:* game.players

Coding and Syntax

console.log(**game.players**)

In typescript, an object may contain different elements, including:

➢ **methods**

➢ **properties**

➢ *...*

Console: Practice

How many players in the space?

➢ keys() *: a function transform object to array*

➢ .length *: a property of an array informing the number of elements*

➢ `console.log(keys(game.players).length);`

```
> console.log(keys(game.players).length);
  2
```

Thank you very much!

Q&A

Prof. Dr. & Dipl. Des. Wei-Chi Chien

Lab of Humanistic Experience and Technology Design

xtdesign.org

chien@xtdesign.org

像素惡地進階使用手冊

Pixel Badlands Advanced User Manual

Event Subscription

Prof. Dr. & Dipl. Des. Wei-Chi Chien

Lab of Humanistic Experience and Technology Design| NCKU

Review: To Our Coding Practice in this Course

✓   *programming language: JavaScript, TypeScript*

learning goal:

⊞   *use online reference to program your interactive mechanism*

✓   *use browser **console** to run your code and interact with the **API***

⊞   *with fundamental know, learn coding on your own*

⊞   *properly express your technical problems*

⊞   *finish a complex goal by smaller steps*

for Today

- ⊞ Declaration
  - Variable Declaration
  - Function Declaration

- ⊞ Event Subscription
  - Event
  - subscribeToEvent()
  - Unsubscribe

- ⊞ If-Statement

- ⊞ Get Information
  - context.player

- ⊞ Make Reaction
  - game.ghost()
  - game.moveMapObject()
  - game.teleport()

Syntax: Variable Declaration

(human language)

There is a number.

(computer language)

```
let number;
      ( or
var number; )
```

// declaration

Syntax: Value Assignment

(human language)

This is a number. The number is 11.

(computer language)

```
let number = 11;
```

or

```
let number;
```

```
number = 11;
```

// initialization by an assignment

Practice

⊞ console.log(myVariableA);

⊞ let myVariableB;
   console.log(myVariableB);

⊞ let myVariableC;
   myVariableC=3;
   console.log(myVariableC);
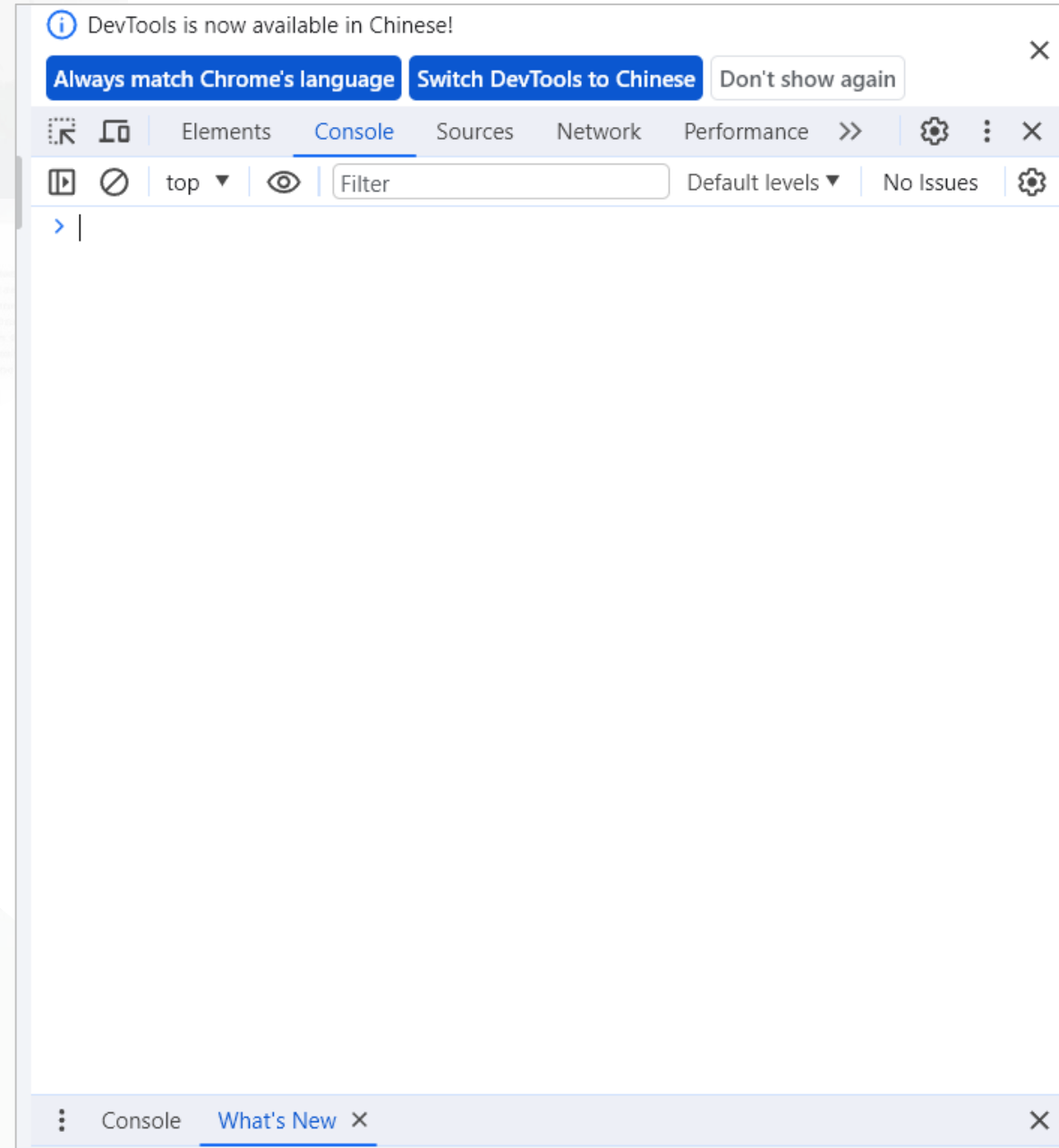
Syntax: Function (功能/函式) Declaration

(human language)

To buy milk, one has to go to the supermarket and buy the milk.

(computer language)

```
function buy(something, somewhere) {
    goto(somewhere);

    buy(something);

}

buy(milk, supermarket);
```

function declaration

call function

// function declaration

Practice

⊞ *We want to declare function to present a given user's name in a greeting.*

⊞
```
function showName(myName){
    console.log("Nice to meet you.");
    console.log("My Name is " + myName + ".");
}
```

⊞ `showName("Jack");`

⊞ `showName("Lucy");`

An Event

✦ *When a user acts her/his avatar on gather.town, one or more properties of this space changes, which is defined as "events" in typeScript.*

console	Reaction		user/player
	property change **Event**	*Space*	
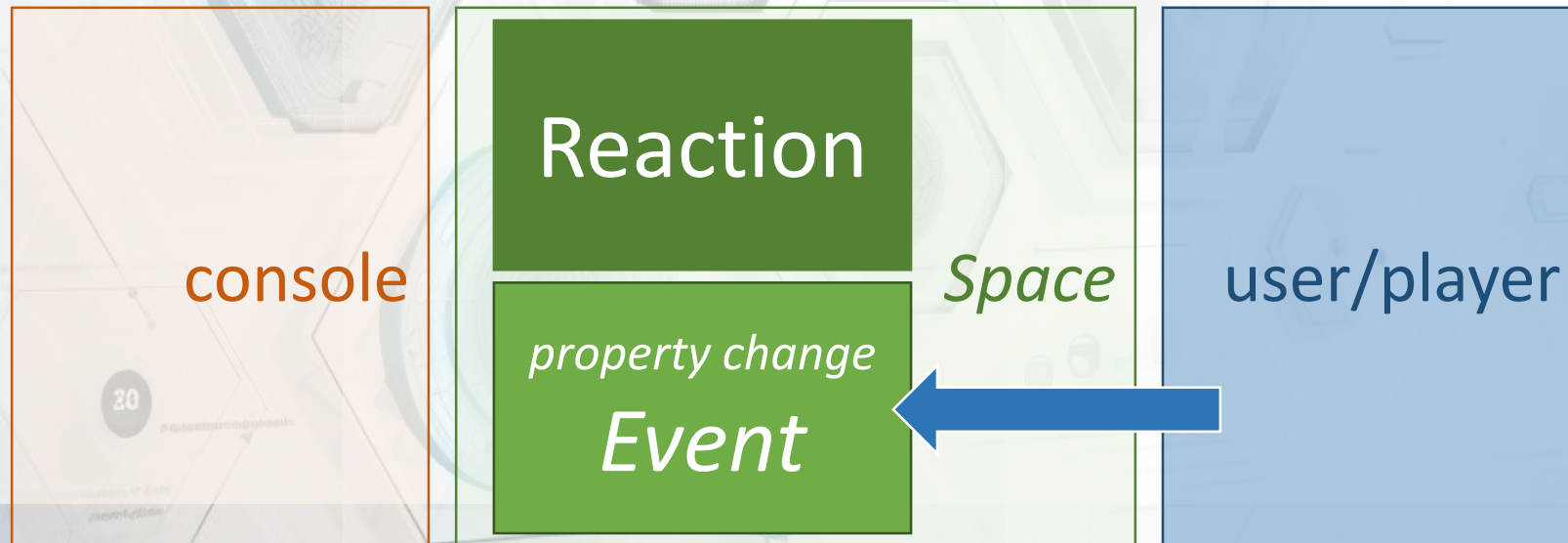
An Event

⊞ *By **subscribing** (listening to / detecting) these events and defining **callback function** to perform reactions, we can create a mechanism of reactions to interact with the user.*

console — callback function → **Reaction** → user/player

Reaction → *Space*

property change **Event** ← user/player
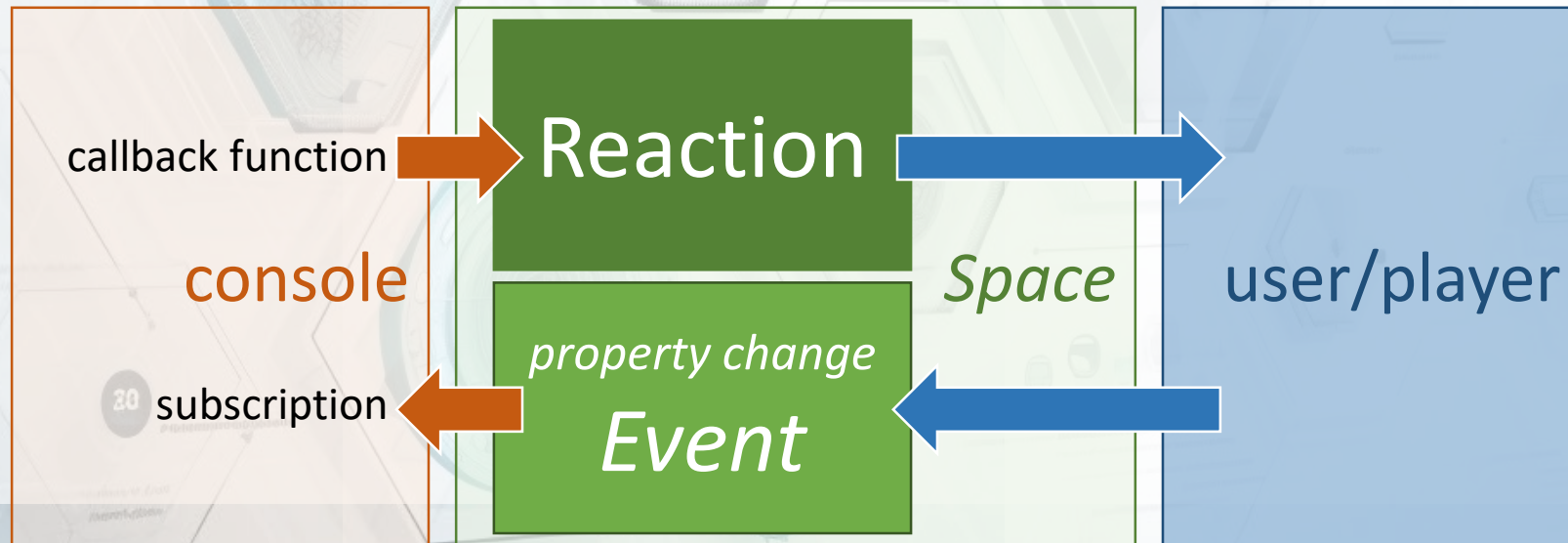
subscription ←

Syntax: Event Subscription

game.subscribeToEvent(*EVENT_TYPE*, *CALLBACK_FUNCTION*);

- game.subscribeToEvent(…);
  is a **method** provided by gather.town API's engine (**game object**).

- *EVENT_TYPE*: a String you can find in the event list, defining the events that we can subscribe.

- *CALLBACK_FUNCTION* (回呼函式): a set of reactions that will be triggered when the event is detected.

SUBSCRIPTION FUNCTION	→	CALLBACK FUNCTION
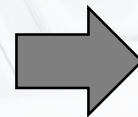
- *You need to declare your own callback function.*

An Event

⊞ *When the hero enters the valley, the dragon starts breathing fire.*

```
subscribe ('enterValley', dragon.fireBreathing() );
```

event   function

Syntax: Callback Function

```
function callbackFunction (data, context) {
    // do something
}


game.subscribeToEvent(EVENT_TYPE, callbackFunction);
```

Practice: Detecting a Ghost

⊞ We can use 'playerGhosts' event-type to detect if any player is ghosting himself on the map.

⊞ If there is a ghost player, show a message in the console.

```
function showGhost (data, context) {
    console.log("There is a ghost!");
}

game.subscribeToEvent('playerGhosts', showGhost);
```

Syntax: Anonymous Function Declaration

(human language)

To buy milk, one has to go to the supermarket and buy the milk.

(computer language)

```
function buy(something, somewhere)     {
     goto(somewhere);

     buy(something);

}
```

function declaration

```
buy(milk, supermarket);
```

call function

// function declaration

Syntax: ~~Anonymous~~ Function Declaration

(human language)

To buy milk, one has to go to the supermarket and buy the milk.

(computer language)

function declaration

```
~~function buy~~(something, somewhere) => {
    goto(somewhere);

    buy(something);

}

buy(milk, supermarket);
```

call function

// function declaration

Syntax: Anonymous Function Declaration

*In some cases, we like to use "**arrow function**" style to declare a function.*

(computer language)

```
(something, somewhere) => {
    goto(somewhere);
    buy(something);
}
```

// function declaration

Callback Function as Anonymous Function

```
function showGhost (data, context) {
    console.log("There is a ghost!");
}


game.subscribeToEvent('playerGhosts', showGhost);
```

Syntax: Callback Function

```
var callbackFunction = (data, context) => {
    ...
}
```

```
game.subscribeToEvent(EVENT_TYPE, (data, context) => {
    console.log("There is a ghost!");
} );
```

Practice: Subscribe to Event

```
game.subscribeToEvent('playerGhosts',
(data, context) =>{
    console.log('There is a ghost!');
});
```

{playerEntersWhisperV2: {…}, p
ves: {…}, mapSetObjectsV2: {…}
▶ accessRequestRespondedTo: {}
▶ accessRequestsUpdated: {7fbb
▶ customEvent: {08b136d8-bb20-
▶ dynamicGates: {1c3bfa23-eddf
▶ fxShakeCamera: {fb5efd82-856
▶ fxShakeObject: {b0bee001-178
▶ mapDeleteObjectByKey: {3b436
▶ mapMoveObject: {a6a10b1b-60f
▶ mapSetAnnouncer: {e6ee81d5-f
▶ mapSetAreas: {f55a7764-777a-
▶ mapSetBackgroundImagePath: {
▶ mapSetDimensions: {e5e5ef0d-
▶ mapSetEnabledChats: {90fd82b
▶ mapSetForegroundImagePath: {
▶ mapSetInitialInfo: {}
▶ mapSetNooks: {78d6c371-379e-
▶ mapSetObjectsV2: {449ee056-b
▶ playerActivelySpeaks: {a2eaf
▶ playerBlocks: {4189f721-890d
▶ playerChats: {c65f59fc-934e-
▶ playerEditsChatMessage: {069
▶ playerEntersPortal: {73c1404
▶ playerEntersWhisperV2: {bf46
▶ playerExits: {09cb5ce5-a9ab-
▶ playerGhosts: {ca4c193d-5923
▶ playerHighFives: {a1f55603-a
▶ playerJoins: {43fded99-4e9b-
▶ playerLeavesWhisper: {16b036
▶ playerMoves: {498b7949-2128-
▶ playerRequestsToLead: {2ef24

More Events

Try the following command lines.

➢ game.subscriptions

➢ *There is plenty of events that we may subscribe.*

➢ *Not all events are available in gather.town.*

→*Play with the list!*

{playerEntersWhisperV2: {…}, playerLeavesWhisper: {…}, playerExits: {…}, playerMoves: {…}, mapSetObjectsV2: {…}, …} ⓘ

▶ accessRequestRespondedTo: {}
▶ accessRequestsUpdated: {7fbba134-c86d-469f-abed-9c251be8d14d: {…}}
▶ customEvent: {08b136d8-bb20-4e61-9a0a-8a279d08efe1: {…}, 69cd5c1b-8f85-446a-b53
▶ dynamicGates: {1c3bfa23-eddf-409e-a3dc-f6fe26850c12: {…}}
▶ fxShakeCamera: {fb5efd82-8562-4d72-bea8-50032ad7e2ad: {…}}
▶ fxShakeObject: {b0bee001-178d-408f-85aa-5b3661f2e288: {…}}
▶ mapDeleteObjectByKey: {3b436b80-4526-4776-bd85-10a8f7d47606: {…}, 8a1b2fdd-c3ea
▶ mapMoveObject: {a6a10b1b-60f7-463a-87f7-e7beeec1f863: {…}, 0d8fbc1e-9949-43c9-a
▶ mapSetAnnouncer: {e6ee81d5-fdc5-48e4-90ee-277b75e6b7d6: {…}, 03d99294-f457-4852
▶ mapSetAreas: {f55a7764-777a-474f-93ed-a698fcc463fa: {…}}
▶ mapSetBackgroundImagePath: {ba16e89a-4f10-474a-97dc-ef7947a47817: {…}, dda1dce3
▶ mapSetDimensions: {e5e5ef0d-5da3-49f1-9a3e-9ed021d2f79e: {…}}
▶ mapSetEnabledChats: {90fd82b6-a420-4962-9aa6-a8f5a5ba86b4: {…}}
▶ mapSetForegroundImagePath: {7f9ab7e1-17a9-487c-abc8-c53beda91112: {…}, c905978d
▶ mapSetInitialInfo: {}
▶ mapSetNooks: {78d6c371-379e-484a-9e6f-c1486b937233: {…}, 662d78da-7d7c-4cac-a7c
▶ mapSetObjectsV2: {449ee056-b650-4a90-b4f3-a21c020678c5: {…}, f0d76b6b-98d5-4093
▶ playerActivelySpeaks: {a2eaf00f-1fac-4a13-b2f6-f5e554b5916c: {…}, 57718f38-8b2b
▶ playerBlocks: {4189f721-890d-4a51-8424-5d8481eef064: {…}}
▶ playerChats: {c65f59fc-934e-4a99-a01e-a81920163dd7: {…}, 43bb2d09-4f97-4687-9c4
▶ playerEditsChatMessage: {0690b88a-9d35-4fad-b394-7b17c0a5818c: {…}}
▶ playerEntersPortal: {73c14049-9864-49cc-951c-1bf8f6089bbd: {…}}
▶ playerEntersWhisperV2: {bf469a20-6ffd-4e17-82dc-ea0ad797a20d: {…}, 5f0d9261-c01
▶ playerExits: {09cb5ce5-a9ab-4139-b3ce-5225bc6c27a6: {…}, ff8006cd-6ffe-492d-b7d
▶ playerGhosts: {ca4c193d-5923-457c-9ebd-e6de49f22da9: {…}, 692ed096-6e8d-4bcb-ab
▶ playerHighFives: {a1f55603-acca-42f9-ae3d-ea841a424e83: {…}}
▶ playerJoins: {43fded99-4e9b-411c-bdfe-569a52104c13: {…}, 3aef5eb9-3430-4c45-a31
▶ playerLeavesWhisper: {16b0365f-adea-4ded-a5a7-f7078b4505b9: {…}, f69fffe1-baef-
▶ playerMoves: {498b7949-2128-4463-9485-1baf34f9c7da: {…}, cda9eb19-6cd8-4ef4-882
▶ playerRequestsToLead: {2ef24975-b34e-4e19-b791-7aeba03963b1: {…}}
▶ playerRequestsToLeadV2: {e3183460-ed95-4192-b369-a3c0bb45c971: {…}}
▶ playerRings: {30ccfdf1-4cc3-422e-8b4f-77196d69a23d: {…}}
▶ playerSetsActivityStatus: {8b392aad-6c84-4885-aa61-b4b50341c9cd: {…}}
▶ playerSetsAmbientPlayersMuted: {0dda21c5-4278-4384-946c-e956cc11f377: {…}}
▶ playerSetsAvailability: {56f0cb8b-626b-4c0c-b7b5-295832792ec4: {…}, 860d9788-b9
▶ playerSetsAway: {af498afd-6a11-49c6-8f1f-fc3cce939d0a: {…}}
▶ playerSetsCurrentlyEquippedWearables: {3ab51514-d35a-4d24-80f7-eae4efe5ch45: {…

Console   What's New   Issues

Practice: Ooops! I screwed up!

⌗ *You cannot change the codes that are already running in the console.*

⌗ *Codes running in the console will stop every time when you close the window.*

⌗ *Use a note pad to save the codes you have written.*

⌗ *Refresh the page, if you ruined your map. Use Map Editor to modify your map again if necessary.*

▶ fxShakeCamera: {fb5efd82-8562-4d72-bea
▶ fxShakeObject: {b0bee001-178d-408f-85
▶ mapDeleteObjectByKey: {3b436b80-4526-4
▶ mapMoveObject: {a6a10b1b-60f7-463a-87
▶ mapSetAnnouncer: {e6ee81d5-fdc5-48e4-9
▶ mapSetAreas: {f55a7764-777a-474f-93ed
▶ mapSetBackgroundImagePath: {ba16e89a-
▶ mapSetDimensions: {e5e5ef0d-5da3-49f1
▶ mapSetEnabledChats: {90fd82b6-a420-490
▶ mapSetForegroundImagePath: {7f9ab7e1-1
▶ mapSetInitialInfo: {}
▶ mapSetNooks: {78d6c371-379e-484a-9e6f-
▶ mapSetObjectsV2: {449ee056-b650-4a90-b
▶ playerActivelySpeaks: {a2eaf00f-1fac-4
▶ playerBlocks: {4189f721-890d-4a51-842
▶ playerChats: {c65f59fc-934e-4a99-a01e-
▶ playerEditsChatMessage: {0690b88a-9d3
▶ playerEntersPortal: {73c14049-9864-49
▶ playerEntersWhisperV2: {bf469a20-6ffd-
▶ playerExits: {09cb5ce5-a9ab-4139-b3ce-
▶ playerGhosts: {ca4c193d-5923-457c-9eb0
▶ playerHighFives: {a1f55603-acca-42f9-4
▶ playerJoins: {43fded99-4e9b-411c-bdfe-
▶ playerLeavesWhisper: {16b0365f-adea-4
▶ playerMoves: {498b7949-2128-4463-9485-
▶ playerRequestsToLead: {2ef24975-b34e-4
▶ playerRequestsToLeadV2: {e3183460-ed9
▶ playerRings: {30ccfdf1-4cc3-422e-8b4f-
▶ playerSetsActivityStatus: {8b392aad-6
▶ playerSetsAmbientPlayersMuted: {0dda2
▶ playerSetsAvailability: {56f0cb8b-626b
▶ playerSetsAway: {af498afd-6a11-49c6-8

Return of the Subscribe Function

⊕ What is "return"(回傳/返回)?

⊕ In Arduino,

```
int val = analogRead(A0);
```

⊕ The analoRead() function **returns** the value from the Arduino I/O pin.

⊕ We assign the returned value to the variable val.

⊕ Some functions have return; some do not.

```
analogWrite(8, 128);
```

Return of the Subscribe Function

✚ game.subscribeToEvent() **has a return**.

✚ It returns a special function, which is used to stop the subscription.

```
let stopShowGhost = game.subscribeToEvent('playerGhosts',
(data, context) =>{
        console.log('There is a ghost!');
}

stopShowGhost(); //stop event subscription
```

Practice: Subscribe to Event

Please try to subscribe the following events. Display messages (using console.log() *) when the events are detected.*

⊞ *playerGhosts*

⊞ *playerShootsConfetti*

⊞ *playerMoves*

⊞ *playerInteractsWithObject*

{pLayerEntersWhisperV2: {…}, p
ves: {…}, mapSetObjectsV2: {…}
  ▶ accessRequestRespondedTo: {}
  ▶ accessRequestsUpdated: {7fbb
  ▶ customEvent: {08b136d8-bb20-
  ▶ dynamicGates: {1c3bfa23-eddf
  ▶ fxShakeCamera: {fb5efd82-856
  ▶ fxShakeObject: {b0bee001-178
  ▶ mapDeleteObjectByKey: {3b436
  ▶ mapMoveObject: {a6a10b1b-60f
  ▶ mapSetAnnouncer: {e6ee81d5-f
  ▶ mapSetAreas: {f55a7764-777a-
  ▶ mapSetBackgroundImagePath: {
  ▶ mapSetDimensions: {e5e5ef0d-
  ▶ mapSetEnabledChats: {90fd82b
  ▶ mapSetForegroundImagePath: {
  ▶ mapSetInitialInfo: {}
  ▶ mapSetNooks: {78d6c371-379e-
  ▶ mapSetObjectsV2: {449ee056-b
  ▶ playerActivelySpeaks: {a2eaf
  ▶ playerBlocks: {4189f721-890d
  ▶ playerChats: {c65f59fc-934e-
  ▶ playerEditsChatMessage: {069
  ▶ playerEntersPortal: {73c1404
  ▶ playerEntersWhisperV2: {bf46
  ▶ playerExits: {09cb5ce5-a9ab-
  ▶ playerGhosts: {ca4c193d-5923
  ▶ playerHighFives: {a1f55603-a
  ▶ playerJoins: {43fded99-4e9b-
  ▶ playerLeavesWhisper: {16b036
  ▶ playerMoves: {498b7949-2128-
  ▶ playerRequestsToLead: {2ef24

像素惡地進階使用手冊

Pixel Badlands Advanced User Manual

Q&A

NCKU ICID | ID 2023-2025

Prof. Dr. & Dipl. Des. Wei-Chi Chien

Lab of Humanistic Experience and Technology Design

像素惡地進階使用手冊

Pixel Badlands Advanced User Manual

Create a Complex Event with if-statement

Prof. Dr. & Dipl. Des. Wei-Chi Chien

Lab of Humanistic Experience and Technology Design| NCKU

Ghost Somebody

http://gather-game-client-docs.s3-website-us-west-2.amazonaws.com/index.html

Engine

Game

Player

getPlayerUidFrom Encld

ghost

getPlayersInMap

getStats

ghost 🔗

```
ghost(ghost, targetId?): void
```

Puts the currently connected client in ghost mode

Parameters

- **ghost**: *number*

  ghost value, either 1 or 0; whether you are ghosting or not.

- **Optional** **targetId**: *string*

  id of player whose ghost value you want to set (must be space owner to set)

Returns *void*

The question mark indicates optional parameters.

Ghost Somebody

- Find player's id.

    `game.players`

- Try this:

    `game.ghost(1, "xxxxx");`

```
follow    rget:
ghost: 0
handRaisedAt: ""
id: "eKQ9sXU3lgPCvDXnmT2pwbswBW22"
inConversation: false
▸ inventory: {items: {…}, order: {…}}
```

```
ghost  🔗

ghost(ghost, targetId?): void
```

Syntax: if-statement

(human language)

"If you get more than 90 marks on your test, I'll buy you a new computer. But if you get 60 marks or less, I will take away your old computer."

(computer language)

```
if (score_of_the_test >= 80){
        buy(you, new_computer);
}else if (score_of_the_test < 60){
        takeAway(old_computer
}
```

← ***condition presented in math***

Syntax: If-Statement

(human language)

"If the visitor is Jack, I will let him in."

(computer language)

```
if (visitor === 'Jack'){

    letIn(visitor);

}
```

condition presented in math
Three equal signs describe the equivalence both in value and data type.

Syntax: If-Statement

(human language)

"One can play the game, only if s/he is older than 18 and younger than 60."

(computer language)

if (age > 18 **&&** age <60){

    playGame = accepted;

}

boolean operators
&& : *and*
|| : *or*

data & context

```
let stopSubscribeMove =
game.subscribeToEvent('playerMoves', (data, context) =>{
    console.log(data);
    console.log(context);
} );


stopSubscribeMove();
```

data & context

- let stopSubscribe = game.subscribeToEvent('playerMoves', (data, context) =>{

    console.log(data);

    console.log(context);

  });

- stopSubscribe();

Depending on the type of an event, you will get different 'data' information. You also get other related data in 'context.'
Check those information for your interaction design.

data & context

- let stopSubscribe = game.subscribeToEvent('playerMoves', (data, context) =>{

    console.log(data);

    console.log(context);

  });

- stopSubscribe();

When calling the game.subscribeToEvent() function, it returns (generates) a function for unsubscribing the event.
We can call this function to stop the event subscription.

data & context

⊕ **let stopSubscribeMove =**

```
game.subscribeToEvent('playerMoves', (data, context) =>{
    console.log(data);
    console.log(context);
} );
```

data →

▸ {$case: 'playerMoves', playerMoves: {…}}   VM1369:2

                                                     VM1369:3

context →
▸ {spaceId: 'y1rNubRsmnM5navp\\SandBox', player: {…}, playerId: 'eKQ9sXU3LgPCvDXnmT2pwbswBW22'}

data & context

```
                                                    VM1369:3
{spaceId: 'y1rNubRsmnM5navp\\SandBox', player: {…}, playerId:
'eKQ9sXU3lgPCvDXnmT2pwbswBW22'} i
  player:
    action: "dismount"
    activelySpeaking: 0
    activityStatus: {}
    affiliation: ""
    allowScreenPointer: true
    ambientPlayersMuted: {}
    away: true
    city: ""
    connected: true
    country: ""
    currentlyEquippedWearables: {skin: 'jPpCpaIsSdSDnNIY6MOQ',
    description: ""
    deskInfo: {deskId: '', locked: false, mapId: ''}
    direction: 5
    displayEmail: ""
    emojiStatus: ""
    emote: ""
    eventStatus: ""
    focusModeEndTime:
    followTarget: ""
    ghost: 0
    handRaisedAt: ""
    id: "eKQ9sXU3lgPCvDXnmT2pwbswBW22"
    inConversation: false
    inventory: {items: {…}, order: {…}}
    isAlone: true
    isInFocusMode: false
    isMobile: false
    isNpc: false
    isSignedIn: true
    itemString: ""
    lastActive: "2025-04-01T15:39:12.993Z"
```

```
    lastInputId: 35
    lastRaisedHand: ""
    lastVisited: ""
    manualVideoSrc: ""
    map: "IHbvf4cRABIEFE7ADvni
    name: "CHI"
    personalImageId: ""
    personalImageUrl: "https://
    phone: ""
    previousVehicleId: ""
    profileImageId: ""
    profileImageUrl: "https://
    pronouns: ""
    role: "Guest"
    shimmy: false
    spaceUserUuid: "ae2cc0b2-3
    speedModifier: 1
    spotlighted: 0
    statusExpiresToExpireTime:
    statusExpiresToStatus: "Do
    statusUpdatedAt: ""
    subtitle: ""
    textStatus: ""
    timezone: ""
    title: ""
    userUuid: "7b737e43-11cd-4
    vehicleId: ""
    vehicleId: ""
    whisperId: ""
```

```
                                    VM1369:2
{$case: 'playerMoves', playerMoves: {…}} i
  $case: "playerMoves"
  playerMoves:
    direction: 5
    encId: 1
    lastInputId: 35
    x: 26
    y: 21
    [[Prototype]]: Object
  [[Prototype]]: Object
```

```
handRaisedAt: ""
id: "eKQ9sXU3lgPCvDXnmT2pwbswBW22"
inConversation: false
```

data →

使用者的x, y座標
data.playerMoves.x

使用者的id
context.player.id

context →

data & context

```
let stopSubscribeMove = game.subscribeToEvent('playerMoves',
(data, context) =>{
    if ( (data.playerMoves.x + data.playerMoves.y)%2 !=0 ){

        game.ghost(1, context.player.id);

    }else{

        game.ghost(0, context.player.id);

    }

});
```

```
▼ {$case: 'playerMoves', playerMoves: {…}} ⓘ
    $case: "playerMoves"
  ▼ playerMoves:
        direction: 5
        encId: 1
        lastInputId: 35
        x: 26
        y: 21
      ▶ [[Prototype]]: Object
    ▶ [[Prototype]]: Object
```

```
handRaisedAt: ""
id: "eKQ9sXU3lgPCvDXnmT2pwbswBW22"
inConversation: false
```

tips: Use *indent*

```
let stopSubscribeMove = game.subscribeToEvent('playerMoves',
(data, context) =>{
    if ( (data.playerMoves.x + data.playerMoves.y)%2 !=0 ){

        game.ghost(1, context.player.id);

    }else{

        game.ghost(0, context.player.id);

    }

});
```

We like to use indent to visually organize our code.

if…else Abbreviation – ternary statement

```
let stopSubscribeMove = game.subscribeToEvent('playerMoves',
(data, context) =>{
    if ( (data.playerMoves.x + data.playerMoves.y)%2 !=0 ){
        game.ghost(1, context.player.id);
    }else{
        game.ghost(0, context.player.id);
    }
});
```

if…else Abbreviation – ternary statement

```
let stopSubscribeMove = game.subscribeToEvent('playerMoves',
(data, context) =>{
    (data.playerMoves.x + data.playerMoves.y)%2 !=0 )?

        game.ghost(1, context.player.id)

    :

        game.ghost(0, context.player.id);

});
```

if…else Abbreviation – ternary statement

```
let stopSubscribeMove = game.subscribeToEvent('playerMoves',

(data, context) =>{

    (data.playerMoves.x + data.playerMoves.y)%2 !=0 )?

game.ghost(1, context.player.id) : game.ghost(0, context.player.id);

});
```

⌗ only for simple if- / if…else statement!!

Find More Properties

⊞ console.log(data) / condole.log(context) in a subscribe() method

⊞ game.players;

⊞ gameSpace.getMyPlayerMap();

Practice

⊞ When a player shoots confetti, s/he is teleported to the location (0, 0).

hint:

⊞ What is the event to be subscribed?

⊞ Check `teleport()` method on the reference page.

⊞ Use `gameSpace.getMyPlayerMap()` to find your map id.

像素惡地進階使用手冊

Pixel Badlands Advanced User Manual

Q&A
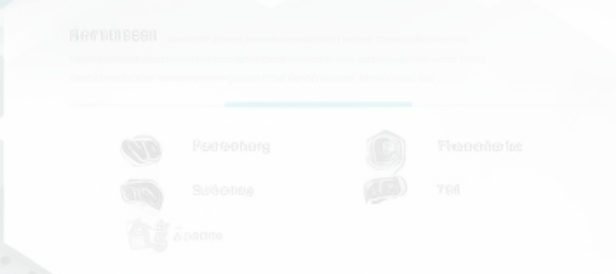
NCKU ICID | ID 2023-2025

Prof. Dr. & Dipl. Des. Wei-Chi Chien

Lab of Humanistic Experience and Technology Design

像素惡地進階使用手冊

Pixel Badlands Advanced User Manual

Design a Mechanism

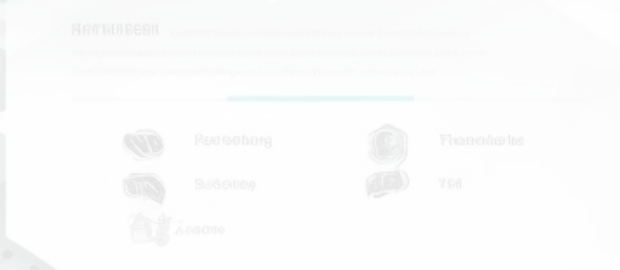Prof. Dr. & Dipl. Des. Wei-Chi Chien

Lab of Humanistic Experience and Technology Design| NCKU

Practice

- When a player shoots confetti, s/he is teleported to the location (0, 0).

hint:

- What is the event to be subscribed?

- Check `teleport()` method on the reference page.

- Use `gameSpace.getMyPlayerMap()` to find your map id.

Practice

⊕ When a player shoots confetti, s/he is teleported to a random location.

hint:
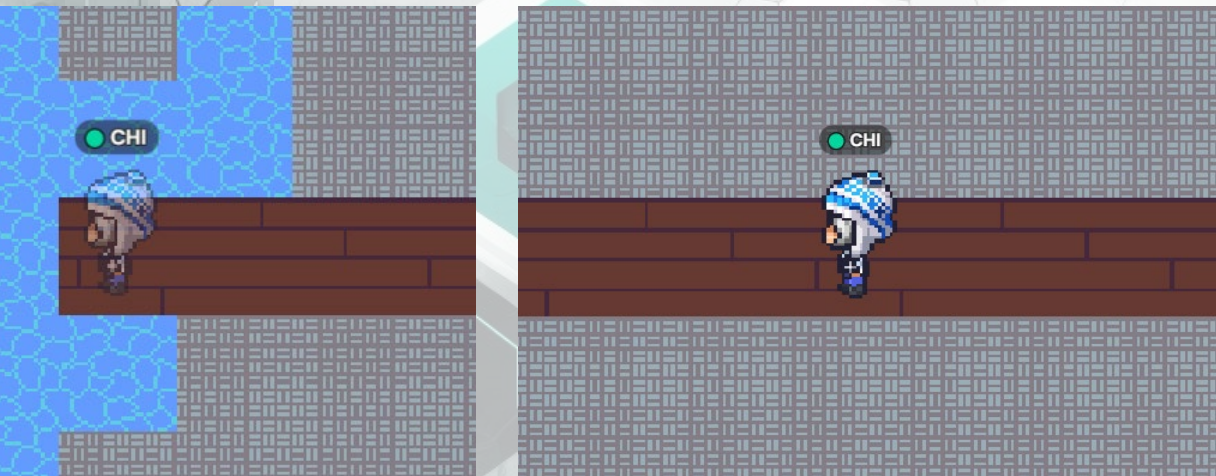
⊕ Use Math.random() to generate a number between 0 and 1.

How to Design an Mechanism on Gather?

| event to be subscribed | → | find the required properties | → | LOGIC | → | method for resulted behavior |

hint

- ⌗ Use `game.eventSubscriptions()`
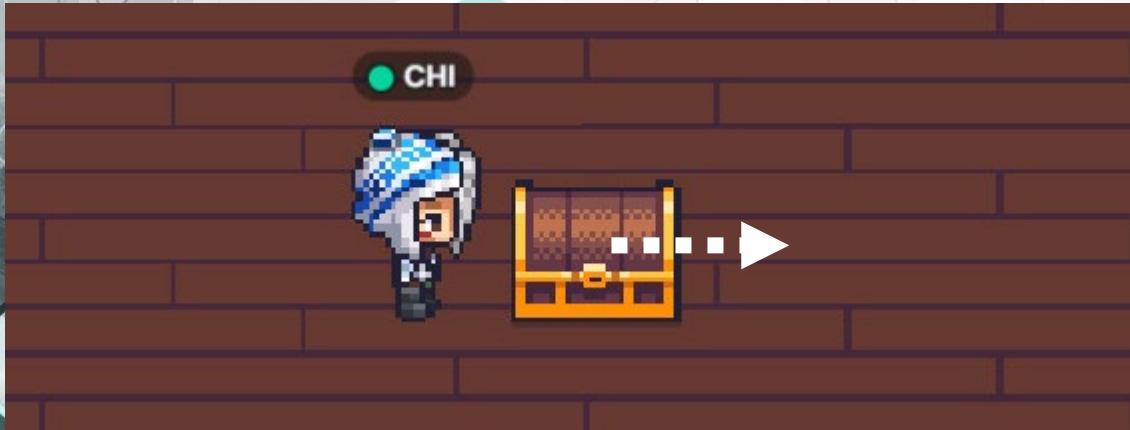- ⌗ Use the reference page.

Practice: Find a Way Out

"The player is walking on a endless path, and there is no way out. Only if the player is 'ghost,' s/he can reach the end of the path."

⊞ What is the event to be subscribed?

⊞ What is the method used to make the reaction behavior?

Practice: We Want to Push an Object

"If a player stands next to an object on the left side, the object will be pushed one tile space to right."

⊞ What is the event to be subscribed?

⊞ What is the method used to make the reaction behavior?

Practice: The Ghost is behind Me

"No matter where the player goes, a ghost is always behind him/her."

- ⊞ What is the **event** to be subscribed?

- ⊞ What is the **method** used to make the reaction behavior?

像素惡地進階使用手冊

Pixel Badlands Advanced User Manual

Q&A

NCKU ICID | ID 2023-2025

Prof. Dr. & Dipl. Des. Wei-Chi Chien

Lab of Humanistic Experience and Technology Design