

Javascript OOP

Lecturer: Resnick Chang
resnick1223@gmail.com



想一想資料的表達

資料是什麼？

- 如何描述一個學生
 - 通常可能有姓名，學號，生日
 - 通訊方式可能有住址，email，聯絡電話
 - 聯絡電話可能有手機，住家電話與辦公室電話

把資料寫出來

- 某個學生
 - 姓名： 張元鴻
 - 學號： 12345678
 - 生日： 12/23
 - 聯絡電話
 - 手機： 0912345678

換成英文

- a Student
 - name: 張元鴻
 - id: 12345678
 - birthday: 12/23
 - contact
 - mobile: 0912345678

正式一點

- 文字用雙引號包住
- 數字可以直接寫
- a Student
 - name: "張元鴻"
 - id: "12345678"
 - birthday: "12/23"
 - contact
 - mobile: "0912345678"

練習：描述一個房子

- 想想哪些房子的東西是可以條列出來的？
 - 坪數
 - 房間數
 - 公共設施佔比或坪數
 - 衛浴數
 - 裝潢
 - 地址
 - 有無車位
 - 有無電梯



進入物件導向的世界

物件

- 將剛剛的資料表示為 Javascript 看得懂的格式
- 具體的一筆資料 = 物件

The screenshot shows a Microsoft Excel spreadsheet with a table of data. The table has columns: client, queryid, mark, deviceplatform, deviceid, devicemodel, state, country, querydwelltime, and session. The data is organized into rows, with the first row being a header. The table is displayed in a blue and white striped format. On the right side of the Excel window, the 'Hive Query' pane is open, showing a list of columns and a 'Columns' section where 'querydwelltime' and 'sessionid' are selected. The 'Columns' section also includes a 'Criteria' section with 'Aggregate Grouping' and 'Ordering' options. The 'Hive Query' pane also includes a 'Hive Connection' section and a 'Hive Objects - Tables/View' section.

	client	queryid	mark	deviceplatform	deviceid	devicemodel	state	country	querydwelltime	session
1	8	18:34:20	en-US	Android	Samsung	SCH-I500	California	United States	13.9204007	0
2	23	19:19:44	en-US	Android	HTC	Incredible	Pennsylvania	United States	1.4757422	0
3	23	19:19:46	en-US	Android	HTC	Incredible	Pennsylvania	United States	0.245968	0
4	23	19:19:47	en-US	Android	HTC	Incredible	Pennsylvania	United States	20.3095339	1
5	28	01:37:50	en-US	Android	Motorola	Droid X	Colorado	United States	16.2981668	0
6	28	00:53:31	en-US	Android	Motorola	Droid X	Colorado	United States	1.7715228	0
7	28	00:53:50	en-US	Android	Motorola	Droid X	Colorado	United States	11.6755987	2
8	28	16:44:21	en-US	Android	Motorola	Droid X	Utah	United States	36.9446892	2
9	28	16:43:41	en-US	Android	Motorola	Droid X	Utah	United States	28.9811416	1
10	28	01:37:19	en-US	Android	Motorola	Droid X	Colorado	United States	3468.538966	0
11	30	17:19:36	en-US	RIM OS	RIM	9650	Massachusetts	United States	66.8533378	0
12	30	17:17:18	en-US	RIM OS	RIM	9650	Massachusetts	United States	2.3198876	0
13	30	17:16:40	en-US	RIM OS	RIM	9650	Massachusetts	United States	1.7547729	1
14	30	00:44:46	en-US	RIM OS	RIM	9330	Massachusetts	United States	857.1453275	1
15	43	00:44:41	en-US	RIM OS	RIM	9330	Massachusetts	United States	12.4195326	0
16	43	21:24:03	en-US	Android	Samsung	SCH-I500	California	United States	0.4996229	0
17	45	21:09:43	en-US	Android	Samsung	SCH-I500	Illinois	United States	1.1773128	0
18	45	20:01:50	en-US	Android	Samsung	SCH-I500	New Jersey	United States	7.7791862	0
19	49	03:05:50	en-US	Android	LG	V5740	New York	United States	39.4991038	0
20	59	01:23:42	en-US	Android	LG	V5660	Nevada	United States	2.3677288	0
21	59	01:23:45	en-US	Android	LG	V5660	Nevada	United States	2.517187	0
22	59	01:23:42	en-US	Android	LG	V5660	Nevada	United States	3.7681792	0
23	59	01:23:42	en-US	Android	LG	V5660	Nevada	United States		
24	62	03:07:36	en-US	Android	LG	V5910	California	United States		
25	62	03:08:15	en-US	Android	LG	V5910	California	United States		
26	62	03:08:17	en-US	Android	LG	V5910	California	United States		

利用變數將資料存起來

- 例如：
學生有姓名，電話的欄位，程式碼像這樣

```
studentName1 = "ABC";  
studentPhone1 = "0912345678";  
studentName2 = "DEF";  
studentPhone2 = "0987654321";
```

物件的表示法

```
var obj = { };
```

大括號代表物件

Property/屬性

- 屬性就是資料的欄位
- 例如前面的
 - 學生的電話
 - 學生的生日
 - 學生的地址
 - 學生的學號

將腦中的屬性寫成程式

- 學生的姓名
主詞 ... 屬性
- 去掉「的」，然後將屬性換行並且縮排
- 例如：
學生
○ 姓名

好像似曾相識的object literal

```
{  
  屬性      值  
  property1 : value1,  
  property1 : value2,  
}
```

Property

- 文字，數字，字串皆可當為屬性
- 任何東西都可以是屬性的值：
 - 物件、陣列、function、數字、文字

```
var obj = {  
  property_1: value_1, // property_# 屬性可以是變數名稱...  
  2: value_2, // 或者是數字...  
  // ..., // 或者什麼都沒有  
  "property n": value_n // 還可以是字串  
};
```

所以某個學生可以表示為

- `var aStudent = {
 name: "張元鴻",
 id: "12345678",
 birthday: "12/23",
 contact: {
 mobile: "0912345678"
 }
};`

例如

```
var student1 = {  
  name: "ABC",  
  phone: "0912345678",  
  city: "taipei",  
  sayHello: function() {  
    document.write("My name is ", this.name);  
  }  
}  
student1.name;  
student1.phone;  
student1.city;  
student1.sayHello();
```

如果有第二個學生

```
var student2 = {  
  name: "DEF",  
  phone: "0987654321",  
  city: "taipei",  
  sayHello: function() {  
    document.write("My name is ", this.name);  
  }  
}  
  
student2.name;  
student2.phone;  
student2.city;  
student2.sayHello();
```

每次都要寫一次



物件的樣板：類別

用function定義類別

```
function ClassName(  
    property1_value, property2_value, property3_value)  
{  
    this.property1 = property1_value;  
    this.property2 = property2_value;  
    this.property3 = property3_value;  
}
```

使用物件

```
var obj = new ClassName(property1_value, property2_value, property3_value);  
console.log(obj.property1);  
console.log(obj.property2);  
console.log(obj.property3);  
obj.MethodName(some input);
```

例如

```
function Student(name, phone, city) {  
    this.name = name;  
    this.phone = phone;  
    this.city = city;  
}
```

```
student1 = new Student("ABC", "0912345678", "Taipei");  
student1.sayHello();  
student2 = new Student("DEF", "0987654321", "Taipei");  
student2.sayHello();
```



物件的動作: 方法

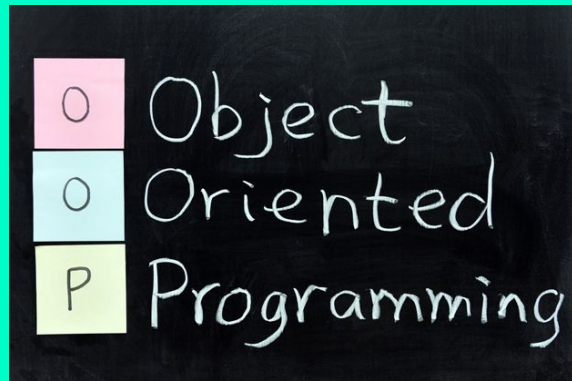
物件的方法: 物件可以執行的動作

- 學生可以跟你打招呼
 - 用英文來表達動作 `say hello`
- 動作等於方法等於 `function`:
 - 第一個單字小寫 `say`
 - 第二個單字大寫 `Hello`
 - 去除中間空白
 - `sayHello`: 這是駝峰命名法
 - `sayHello: function() { }` 這是你執行的函式

用prototype定義類別的屬性與方法

```
ClassName.prototype.MethodName(inputParameter) {  
    //do something here  
}
```

```
function Student(name, phone, city) {  
    this.name = name;  
    this.phone = phone;  
    this.city = city;  
}  
  
Student.prototype.sayHello = function() {  
    document.write("Hello, My name is " + this.name + "<br>");  
}  
  
student1 = new Student("ABC", "0912345678", "Taipei");  
student1.sayHello();  
student2 = new Student("DEF", "0987654321", "Taipei");  
student2.sayHello();
```



物件導向基礎: 繼承

定義一個Person

- Person應該有一些基本資料
 - 姓名
 - 性別
 - 打招呼的方法

再定義一次學生

- 學生有基本資料
 - 姓名
 - 性別
 - 生日
 - 地址
 - 連絡電話
- 好像有些資料重複

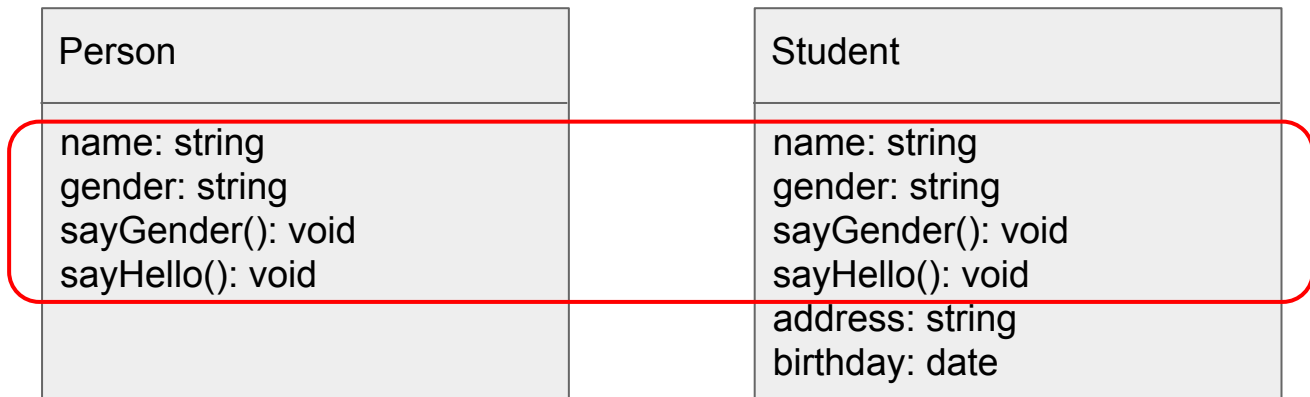
重複的東西有什麼壞處？

- Person類別
 - 追加東西，Student也要跟著追加
 - 刪除東西，Student也要刪除東西
- 但是 but
 - 你忘了追加
 - 你同事忘了追加
- Person跟Student少了共同應該追加的東西
- 所以你就糟糕了!!!

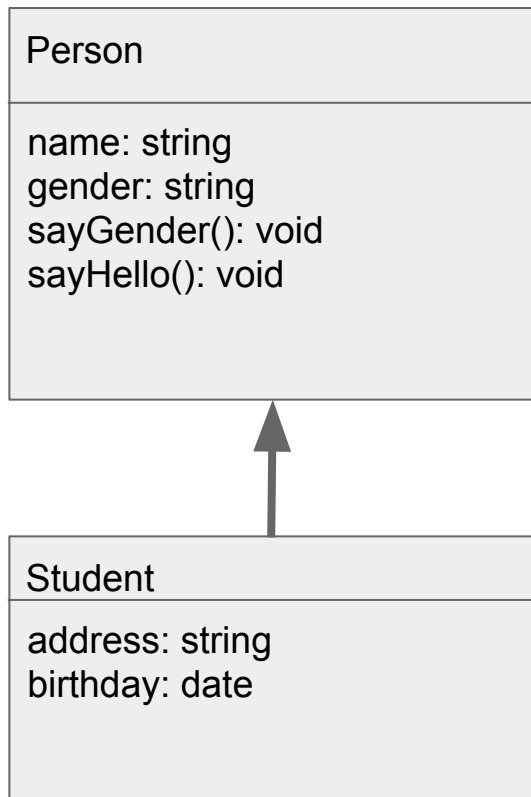
你應該會這樣想

- 所有的學生都是人吧
- 所以人寫過的程式應該不用再寫一次
- 如果需要追加東西或刪除東西就在某個底稿追加就好
- 學生應該跟一般人打招呼用的方法不太一樣
- 如何改寫？

用張圖來呈現想法



用「繼承」來解決



Parent(supertype)與Child

- Person是Parent
- Student是Child
- Child要去繼承Parent擁有的東西
- Child要去追加Parent所沒有的東西
- Parent類別又稱為supertype (超型態)

誰該是Child, 誰該是Parent

- is-a的檢驗法：
 - student is a person?
 - 用中文更直觀一點：
所有的學生都是人嗎？
是的話：Student 繼承 Person
- 練習：
 - 寫下一個關於動物的繼承鏈

原型:prototype

- `prototype`用以描述類別的原始長相
- `className.prototype = {`
 `field1: null,`
 `field2: null,`
 `method1: function() {...},`
 `method2: function() {...}`
 `}`

定義一個類別A

- ```
function A(a) {
 this.varA = a;
}
```
- ```
A.prototype = {  
    varA:null,  
    doSomething: function() {...}  
}
```

定義一個類別B

- ```
function B(a, b) {
 this.varA = a;
 this.varB = b;
}
```
- ```
B.prototype = {  
  varA:null,  
  varB:null,  
  doSomething: function() {...}  
}
```

改寫

- ```
function B(a, b) {
 A.call(this, a);
 this.varB = b;
}
```
- ```
B.prototype = {  
    varB:null,  
    doSomething: function() {  
        A.prototype.doSomething.apply(this, args);  
    }  
}
```

透過傳播原型來繼承

- 繼承 = 透過prototype來取用已經寫好的東西
 - `Student.prototype = new Person();`
 - `Student.prototype.constructor = Student;`
- 翻成中文：
 - Student的原型來自A的原型
 - 將Student的建構式指向Person

練習

- 定義一個Employee類別包含：
 - name, baseSalary兩個欄位
 - 提供getSalary方法取出薪資
- 定義一個Sales與Manager為Employee的子類別
 - Sales增加業績欄位
 - Manager增加績效欄位
 - Sales的薪資等於底薪+業績乘3%
 - Manager的薪資等於底薪+績效成績換成獎金



Homework

使用Object完成

1. Number Guessing 遊戲
2. Online Booking System

