前端應用技術開發
jQuery

王緯宸 Eddie Wang

課程大綱

1. jQuery 簡介
2. jQuery 選擇器
3. jQuery Tree Traversal
4. 事件處理
5. 元素內容與屬性的處理
6. jQuery Animate
7. 附錄

lesson 1

jQuery 簡介

jQuery 簡介

1. 什麼是 jQuery
2. 瀏覽器的支援
3. 如何使用 jQuery
4. 文件備妥處理器

什麼是jQuery

- jQuery 是一個功能豐富的 JavaScript 函式庫
- 由 John Resig 在2006年1月的 BarCamp NYC 上釋出第一個版本。
- jQuery 的口號是「寫得更少，做得更多」，因為它將許多常見的 JavaScript 任務封裝在簡單的方法中，使得開發者可以更快速地編寫代碼。
- 可以讓你在頁面中運用它所提供的選擇器 (Selector) 來選取元件 (element) ，再針對這些被選取的元件，利用 jQuery 提供的功能 (API) 做一些操作或變更。
- 萬能的錢字符 「$」

常用jQuery處理的功能?

- 找尋 HTML 頁面 (HTML document) 中的元素 (element)
- 改變 HTML 的內容
- 根據使用者的操作做出對應的反饋
- 在頁面中使用動畫效果
- AJAX 使用，避免重新整理畫面

瀏覽器的支援

電腦瀏覽器 Desktop

- Chrome: 目前版本及前一版本
- Edge: 目前版本及前一版本
- Firefox: 目前版本及前一版本, ESR 企業版本
- Safari: 目前版本及前一版本
- Opera: 目前版本

行動裝置瀏覽器 Mobile

- Android 4.0 以上的內建瀏覽器
- iOS 7 以上的 Safari

如何使用jQuery?

- 下載 jQuery
- CDN 的引用
- 文件備妥處理器
- jQuery 的使用初體驗

下載jQuery

- 標準版（v1,v2,v3）
  - 正常版（Uncompressed）：做為開發或除錯用。（可讀性高）
  - 壓縮版（Minified）：由正常版去掉註解、換行符號、空白...等等，作為正式上線用。（可讀性低）
  - 映射版（Map）：壓縮版的JavaScript Code對應到正常版的原始碼。
- 精簡版（v3,不支援ajax及effects）
  - 正常版（Uncompressed）：jquery-xxx.slim.js
  - 壓縮版（Minified）：jquery-xxx.slim.min.js
  - 映射版（Map）：jquery-xxx.slim.min.map
- 使用方式
  - <script src="jquery檔案路徑"></script>

CDN方式引用jQuery

- 內容傳遞網路 CDN（Content Delivery Network）是一種分布式的伺服器網絡，幫助快速、安全地傳送網路內容（如圖片、影片、CSS、JavaScript 等）到使用者端。
- jQuery的CDN引用方式
  - <script src="https://code.jquery.com/jquery-3.7.1.js"></script>
- 其他CDN
  - Google CDN
  - Microsoft CDN
  - CDNJS CDN
  - JSDelivr CDN

文件備妥處理器

- 傳統的事件處理中有 window.onload 可在網頁載入完畢後觸發，但會有必須等到圖片與外部資源都載入完畢之後才執行的特性。
- 建立HTML DOM TREE後，即可進行操作，不需要等待圖片下載完成
  - ready() 方法

```
<script>
    // 寫法1
    jQuery(document).ready(function(){
        // 網頁載入後立即執行程式
    })
</script>
```

```
<script>
    // 寫法2
    $(function(){
        // 網頁載入後立即執行程式
    })
</script>
```

鏈式語法 chaining

- 多個方法套用在相同選擇器上，讓程式碼更簡潔。
- 當其中一個方法無法正確執行時，後續的方法也不會被執行。
- 語法：
  - $(selector).text().css("屬性","屬性值")

lesson 2

jQuery 選擇器

jQuery 選擇器

1. 基本選擇器 Basic Selector
2. 內容過濾選擇器 Content Filter Selector
3. 子過濾選擇器 Child Filter Selector
4. 表單選擇器 Form Selector
5. 階層選擇器 Hierarchy Selector

基本選擇器 Basic Selector

- 選取所有元素 All selector： $(" * ")
- 使用標籤名稱選取元素 Element Selector： $(" 標籤名稱")
  - $("div") ⇒ 選取文件中所有div元素
- 使用class屬性選取元素 Class Selector： $(" .class 屬性值")
  - $(".target") ⇒ 選取文件中所有是 class="target" 的元素
- 使用id屬性選取元素 ID Selector： $(" #id 屬性值")
  - $("#target") ⇒ 選取文件中擁有 id="target" 的元素
- 選擇所有指定選擇器的合併條件： $("條件1,條件2,...")
  - $("div,span,.target") ⇒ 選取所有div、span及class="target"的元素

內容過濾選擇器 Content Filter Selector

- :contains(text)
  - 選取所有內文含有括弧中指定字串的元素 ⇒ $("p:contains('hello')")
- :has(selector)
  - 選取含有指定選擇器的元素 ⇒ $("p:has(span)")
- :parent
  - 選取不是空的元素 ⇒ $("p:parent")
- :empty
  - 選取空的元素 ⇒ $("p:empty")

子過濾選擇器 Child Filter Selector

- :first-child
  - 選取指定元素下的第一個子元素 ⇒ $("li:first-child")
- :last-child
  - 選取指定元素下的最後一個子元素 ⇒ $("li:last-child")
- :nth-child(index)
  - 選取指定元素下的第 index 個子元素（ index 從1開始計算） ⇒ $("li:nth-child(2)")
- :only-child
  - 選取指定元素下只有單一子元素 ⇒ $("li:only-child")

表單選擇器 Form Selector

- :input
  - 選取所有 input、textarea、select、button 等元素 ⇒ $(":input")
- :button
  - 選取所有 type="button" 及 button 等元素 ⇒ $(":button")
- :text
  - 選取所有 type="text" 元素 ⇒ $(":text")
- :password
  - 選取所有 type="password" 元素 ⇒ $(":password")

表單選擇器 Form Selector

- :radio
  - 選取所有 type="radio" 元素 ⇒ $(":radio")
- :checkbox
  - 選取所有 type="checkbox" 元素 ⇒ $(":checkbox")
- :submit
  - 選取所有 type="submit" 及 button 等元素 ⇒ $(":submit")

表單選擇器 Form Selector

- :checked
  - 選取表單當中被選擇的項目，包含radio、checkbox、option ⇒ $(":checked")
- :selected
  - 選取表單當中被選擇的項目，只有包含option ⇒ $(":selected")
- :disabled
  - 選取表單當中所有身上有 disabled 屬性的元素， ⇒ $(":disabled")

課堂練習

- 建立一個按鈕，當此按鈕被點擊時『.on("click",function(){})』透過 :checked 判斷使用者在多選題中選了幾個選項，並用console.log()顯示使用者選了幾個選項【.length】。
- 建立一個按鈕，當此按鈕被點擊時 『.on("click",function(){})』透過 :selected 將目前被選擇的選項背景色改為粉紅色。
- 承上題，將沒被選中的選項，背景色變回透明。

階層選擇器 Hierarchy Selector

- parent > child (子選擇器 child selector)
  - 選取指定父層為 parent 條件元素並且其子元素符合 child 條件 ⇒ $("#div1 > p")
- parent child (後代選擇器 descendant selector)
  - 選取指定父層為 parent 條件元素其子元素以及後代元素符合 child 條件 ⇒ $("#div1 p")
- prev + next (同層相鄰選擇器)
  - 選取 prev 條件之後相鄰的元素並符合 next 條件 ⇒ $("#div1 + p")
- prev ~ siblings (同層全體選擇器)
  - 選取 prev 條件之後的兄弟層並符合 siblings 條件的所有元素 ⇒ $("#div1 ~ p")

lesson 3

Tree Traversal

Tree Traversal

1. DOM 與 Tree Traversal
2. 過濾方法 Filtering
3. 尋訪子元素的方法
4. 尋訪父元素的方法
5. 尋訪同階層的方法

DOM 與 Tree Traversal

- DOM
  - Document Object Model(文件物件模型)
  - 是一組用來處理HTML及XML文件的API
  - 文件中的所有資料，皆可透過DOM來存取、修改、刪除及新增
- Tree Traversal
  - 透過DOM Tree 的結構，找到要控制的元素

過濾方法 Filtering

- .eq(index)
  - 功能與 :eq() 相同，index 一樣是從0開始 ⇒ $("td").eq(0)
- .first()
  - 功能與 :first 相同 ⇒ $("td").first()
- .last()
  - 功能與 :last 相同 ⇒ $("td").last()
- .not(selector)
  - 功能與 :not(selector) 相同 ⇒ $("td").not(".target")

過濾方法 Filtering

- .filter(selector)
  - 從尋找到的元素中，再篩選出符合選擇器的元素 ⇒ $("td").filter(".target")
- .index()
  - 找到該元素的索引值 ⇒ $("td").index()
- .slice(start [, end])
  - 單獨設定一個數字，第 start 個元素開始以後都選取 ⇒ $("td").slice(6)
  - 設定兩個數字，第 start 個元素開始到第 end 個元素之前都選取
  - (不含 end) ⇒ $("td").slice(3,7)

尋訪子元素的方法

- .children([selector])
  - 取得指定元素的下一層元素，如有條件可以寫在括弧中 ⇒ $("td").children(".circle")
- .find([selector])
  - 找尋指定元素的子元素或是後代元素 ⇒ $("td").find(".innerCircle")

尋訪父元素的方法

- .parent([selector])
  - 取得指定元素的上一層元素，如有條件可以寫在括弧中 ⇒ $(".circle").parent()
- .parents([selector])
  - 取得指定元素的所有父層元素直到文件的根元素「 html元素 」 ⇒ $(".circle").parents()
- .parentsUntil([selector])
  - 取得指定元素的所有父層元素直到碰到 selector 條件的父元素即停止，取得的元素不包含符合 selector 的元素 ⇒ $(".circle").parentsUntil("table")
- .closet([selector])
  - 從自己本身開始向上找尋，找到第一個符合 selector 的元素 ⇒ $(this).closet("tr")

尋訪同階層(兄弟)的方法

- .next([selector])
  - 取得指定元素的後一個元素，如有要設定條件可以寫在括弧中 ⇒ $(this).next()
- .nextAll([selector])
  - 取得指定元素之後的所有元素，如有要設定條件可以寫在括弧中 ⇒ $(this).nextAll()
- .nextUntil([selector])
  - 取得指定元素之後直到 selector 條件之前，不包含符合 selector 元素 ⇒ $(this).nextUntil(".stop")

尋訪同階層(兄弟)的方法

- .prev([selector])
  - 取得指定元素的前一個元素，如有要設定條件可以寫在括弧中 ⇒ $(this).prev()
- .prevAll([selector])
  - 取得指定元素之前的所有元素，如有要設定條件可以寫在括弧中 ⇒ $(this).prevAll()
- .prevUntil([selector])
  - 取得指定元素之前直到 selector 條件之前，不包含符合 selector 元素 ⇒ $(this).prevUntil(".stop")

尋訪同階層(兄弟)的方法

- .siblings([selector])
  - 取的目前選擇器元素的所有兄弟層 ⇒ $(this).siblings()
- .end()
  - 需要回到前一個選取結果的選取器 ⇒ $("this").next().css().end().prev().css()
  - 從 this 元素的後一個元素設定css，回到上一次的元素(this)的前一個元素設定css

lesson 4

事件處理 Event

事件處理 Event

1. 何謂事件
2. 事件基本語法 Event Basics
3. 事件綁定 On Binding Event
4. Event 物件、Event物件的屬性與方法
5. 滑鼠事件 Mouse Events
6. 鍵盤事件 Keyboard Events
7. 表單事件 Form Events
8. 瀏覽器事件 Browser Events

什麼是事件？

- 使用者對網頁內容做的動作。
  - 使用者點擊網頁中的按鈕 ⇒ 滑鼠點擊(click)。
  - 使用者輸入帳號密碼 ⇒ 鍵盤輸入(keydown)。
- 事件發生時需要做對應的事情，稱為事件處理程序(event handler)，通常是一個函數。
- 事件類型
  - 滑鼠事件 Mouse Events
  - 鍵盤事件 Keyboard Events
  - 表單事件 Form Events
  - 瀏覽器事件 Browser Events
  - 文件載入相關 Document Loadings

事件類型 Event Type

滑鼠事件
click
dblclick
contextmenu
mousedown
mouseup
mouseenter
mouseleave
mouseover
mouseout
mousemove

鍵盤事件
keydown
keypress
keyup

表單事件
blur
change
focus
focusin
focusout
select
submit

瀏覽器事件
error
resize
scroll

文件載入相關
load
ready
unload

事件基本語法 Event Basics

- JavaScript
  - element.addEventListener(" [事件種類 type] ", function ( ){ //觸發後要執行的程式 })
- jQuery
  - selector.on(" [事件種類 type] ", function ( ){ //觸發後要執行的程式 })
  - selector.[事件種類 type](function ( ){ //觸發後要執行的程式 }) 【 3.3版本後已廢除 】

事件綁定 On Binding Event

- 單一事件綁定單一事件處理
  - $([selector]).on("[事件種類type]", function(){ //事件處理 })。

    ⇒ $("#btn1").on("click",function(){ //事件處理 })
- 多個事件綁定單一事件處理
  - $([selector]).on(" [事件種類type] [事件種類type] ", function(){ //事件處理 })

    ⇒ $("#bnt1").on("click contextmenu mouseenter", function(){ //事件處理 })

事件綁定 On Binding Event

- 多個事件綁定多個事件處理
  - $([selector]).on({

    [事件種類type] : function(){ //事件處理 },

    [事件種類type] : function(){ //事件處理 },

    })

    ⇒ $("#btn1").on({

    click : function(){ //事件處理 },

    contextmenu: function(){ //事件處理 },

    })

事件綁定 On Binding Event

- 事件動態綁定
  - $([selector1]).on("[事件種類type]", "[selector2]", function(){ //事件處理 })。
  - selector1：父層元素。（不是動態生成）
  - selector2：子層元素，selector1 的子元素，可以由程式動態生成元素。

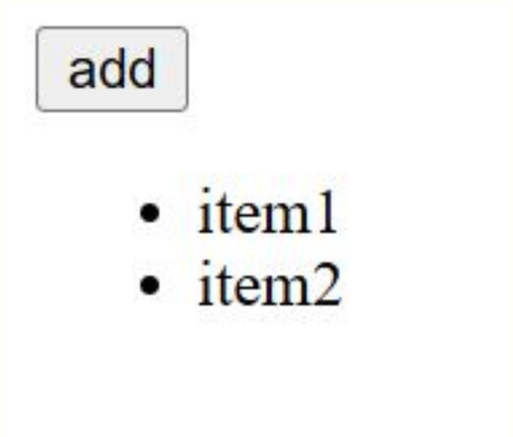
    ⇒ $("#lists").on("click", "li" , function(){ //事件處理 })

事件綁定 On Binding Event

- 只會觸發一次事件
  - $([selector]).one("[事件種類type]", function(){ //事件處理 })。
    ⇒ $("#btn2").one("click",function(){ //事件處理 })
- 移除事件綁定
  - $([selector]).off("[事件種類type]", function(){ //事件處理 })。
    ⇒ $("#off").off("click", btn2Click)

事件動態綁定練習

1. 在畫面上建立一個按鈕『<button>』以及清單 『 』，點擊此按鈕後可以動態建立新的單一項目『<li>』，並且點擊所有單一項目內容皆可以 console.log 該清單之內容。



Event 物件

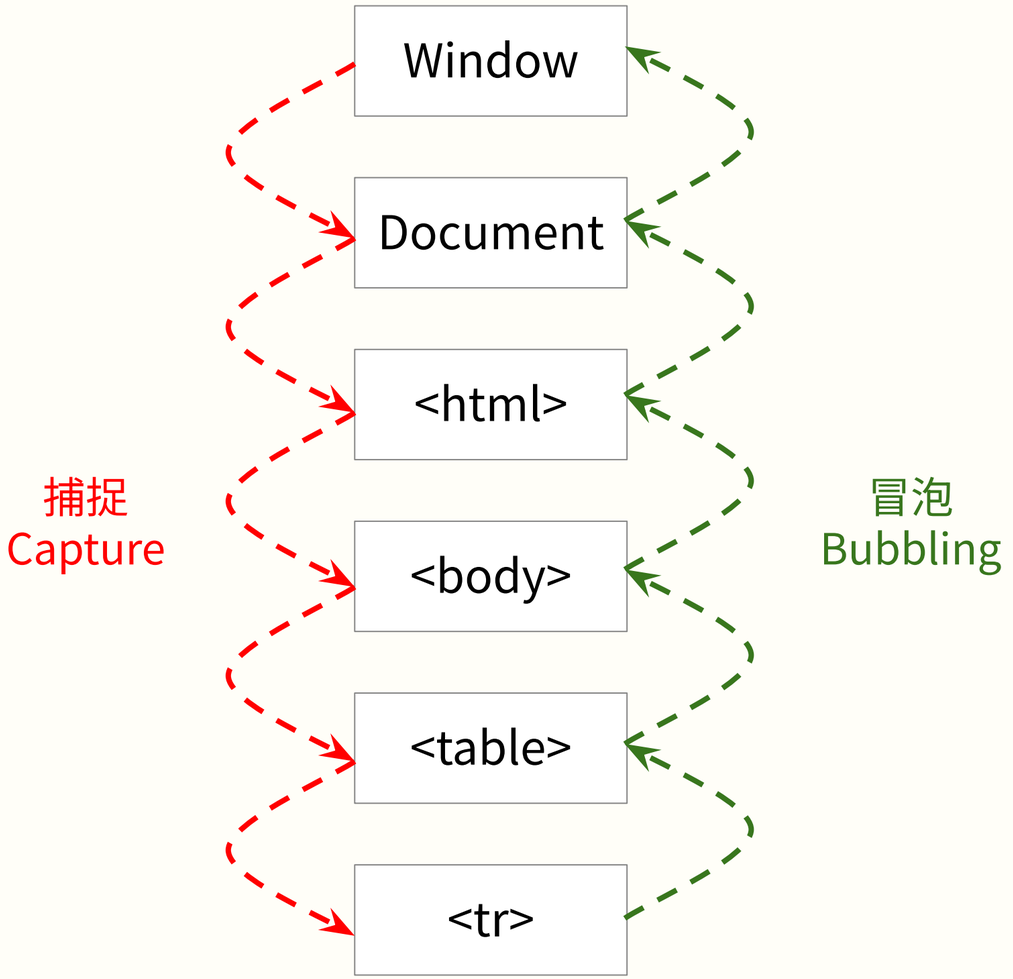
- 何謂 Event 物件
  - 網頁上事件觸發的同時會產生 Event 物件。
  - 由 物件屬性 和 物件方法 組成。

Event 物件的屬性

- type：事件的類型。
- which：回傳鍵盤輸入的編碼或滑鼠按鈕的代號。
- metakey：回傳是否按下 meta 鍵。
- pageX、pageY：滑鼠游標相對於文件左邊界、上邊界的位置。
- target：回傳觸發事件的DOM元素。
- currentTarget：回傳事件正在處理時所在的DOM元素。
- data：回傳data給繫結的事件處理程序。

Event 物件方法

- preventDefault()
  - 停止(取消)預設動作。
  - 例如：超連結預設頁面跳轉功能
  - 例如：input的submit的預設送出表單功能。
- stopPropagation()
  - 取消事件氣泡。
  - 事件的觸發順序是先捕捉再冒泡。
  - addEventListener、jQuery的.on() 預設為冒泡行為。



滑鼠事件 Mouse Events

- .on("click", function)
  - 點擊滑鼠左鍵觸發事件。
- .on("dblclick", function)
  - 雙擊滑鼠左鍵觸發事件。
- .on("contextmenu", function)
  - 點擊滑鼠右鍵觸發事件。
- .on("mousedown", function)
  - 點擊滑鼠按鍵(左右皆可)，不須等待放開按鍵就會觸發事件。
- .on("mouseup", function)
  - 點擊滑鼠按鍵(左右皆可)，等待放開按鍵後才會觸發事件。

滑鼠事件 Mouse Events

- .on("mouseenter", function)
  - 滑鼠游標移入選取的元素時觸發事件。
- .on("mouseleave", function)
  - 滑鼠游標離開選取的元素時觸發事件。
- .on("mouseover", function)
  - 滑鼠游標移入選取的元素與其子元素時觸發事件。
- .on("mouseout", function)
  - 滑鼠游標離開選取的元素與其子元素時觸發事件。
- .on("mousemove", function)
  - 滑鼠游標在選取的元素與其子元素上移動時觸發事件。

滑鼠事件練習

1. 在指定元素中點擊滑鼠右鍵，將預設右鍵清單隱藏。
2. 點擊 【 顯示 】 按鈕將密碼輸入框改為明碼『 .attr() 』並把按鈕文字改為【 隱藏 】『 .text() 』，點擊【 隱藏 】按鈕將密碼輸入框改為預設 * 號 『 .attr() 』並把文字改為【 顯示 】『 .text() 』。

鍵盤事件 Keyboard Events

- .on("keydown", function)
  - 鍵盤按下時觸發事件。
- .on("keyup", function)
  - 鍵盤放開時觸發事件。
- .on("keypress", function)
  - 鍵盤按下又放開時觸發事件。
  - 只針對可以輸出文字、符號的按鍵有效，ESC、Backspace、方向鍵就不會觸發。

鍵盤事件練習

1. 在畫面上建立一個輸入框 『 <input> 』 以及一個段落元素 『 <p> 』 ，偵測使用者在輸入框撰寫內容時，透過鍵盤事件 **" 同步內容 "** 到段落元素中『 .text()、.val() 』。

1-1. 呈上題，修改程式碼，**" 只有使用者按下 Enter 鍵 "**『 event.which == 13 』時才把內容同步顯示到段落元素中。

2. 偵測使用者 **" 同時按下 "** ctrl 及 alt 鍵，才 console.log(" ctrl + alt ")。

表單事件 Form Events

- .on("focus", function)
  - 滑鼠游標移入並可以輸入內容時觸發事件。
- .on("blur", function)
  - 滑鼠游標移出並取消輸入時觸發事件。
- .on("change", function)
  - 下拉式選單選項修改、偵測到輸入控制項 input 內容修改並且游標移開取消輸入時觸發事件。

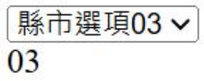
表單事件練習

1. 建立信用卡 16碼 輸入框 『<input>』並限制輸入長度最大為4 『.val().length』， 每輸入4位數後 **" 立即關注 "** 到下一個卡號輸入框 『.next()』。

1-1. 呈上題，當使用者刪除卡號輸入框內的所有數字，**" 立即跳回 "** 上一個卡號輸入框 『.prev()』。

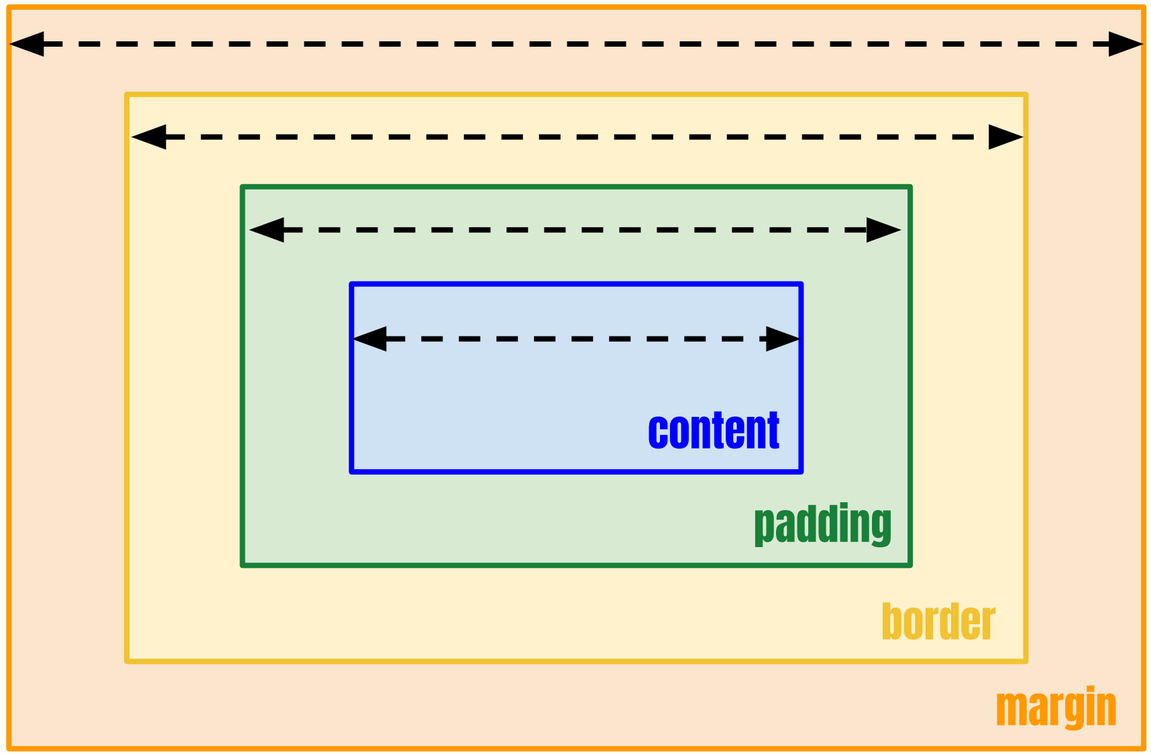
1-2. 如果使用者輸入的是數字以外的內容，將其取代為空 『.replace()』。

2. 建立一個下拉式選單 『 <select> 、<option> 』，當使用者改變下拉式選單，即將該選項的值 『.val()』顯示在『 』中。



視窗大小控制

1. 設定一個容器 『 』，並在其身上加上CSS的padding、border、margin，檢視以上之差異。



outerWidth(true)

outerWidth()

innerWidth()

width()

視窗大小控制

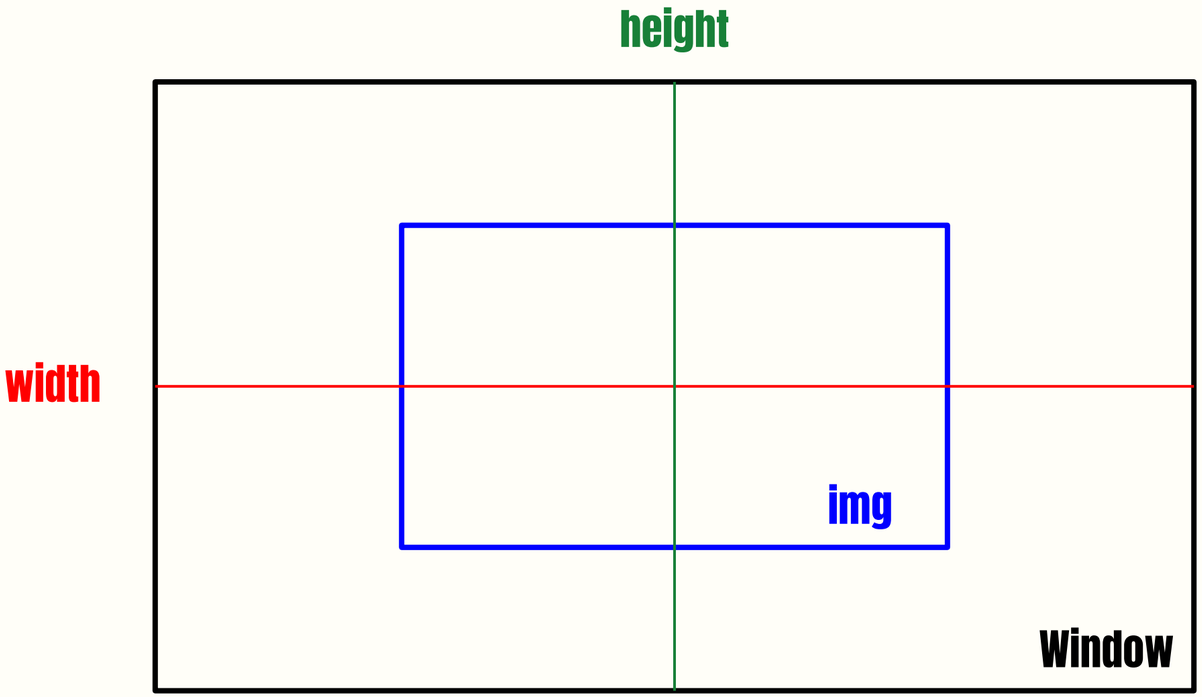
- .on("resize", function)
  - 當瀏覽器的視窗大小改變時觸發事件。
- .width([value])
  - 取得該元素寬度，設定寬度可輸入 value 值。
- .height([value])
  - 取得該元素高度，設定高度可輸入 value 值。

視窗大小控制

- .innerWidth([value])
  - 取得元素寬度，包含 padding 的寬度，設定寬度可輸入 value 值。
- .innerHeight([value])
  - 取得元素高度，包含 padding 的高度，設定高度可輸入 value 值。
- .outerWidth([value], [includeMargin : boolean])
  - 取得元素寬度，包含 padding 與 border 的寬度，設定寬度可輸入 value 值。
  - 如果加上 boolean 值 true 就會包含margin 寬度。
- .outerHeight([value], [includeMargin : boolean])
  - 取得元素高度，包含 padding 與 border 的高度，設定高度可輸入 value 值。
  - 如果加上 boolean 值 true 就會包含 margin 高度。

視窗大小控制練習

1. 透過瀏覽器 『 $(window) 』 的寬度與高度 『 .width()、.height() 』與圖片的寬度高度 **" 相減除以二 "**，每次瀏覽器視窗調整大小時，讓圖片可以在畫面正中間 『 .css() 』。



滾動事件與方法

- .on("scroll", function)
  - 當滑鼠滾動時觸發事件。
- .scrollTop([value])
  - 取得卷軸垂直位置，設定垂直卷軸距離 value 值。
- .scrollLeft([value])
  - 取得卷軸水平位置，設定水平卷軸距離 value 值。

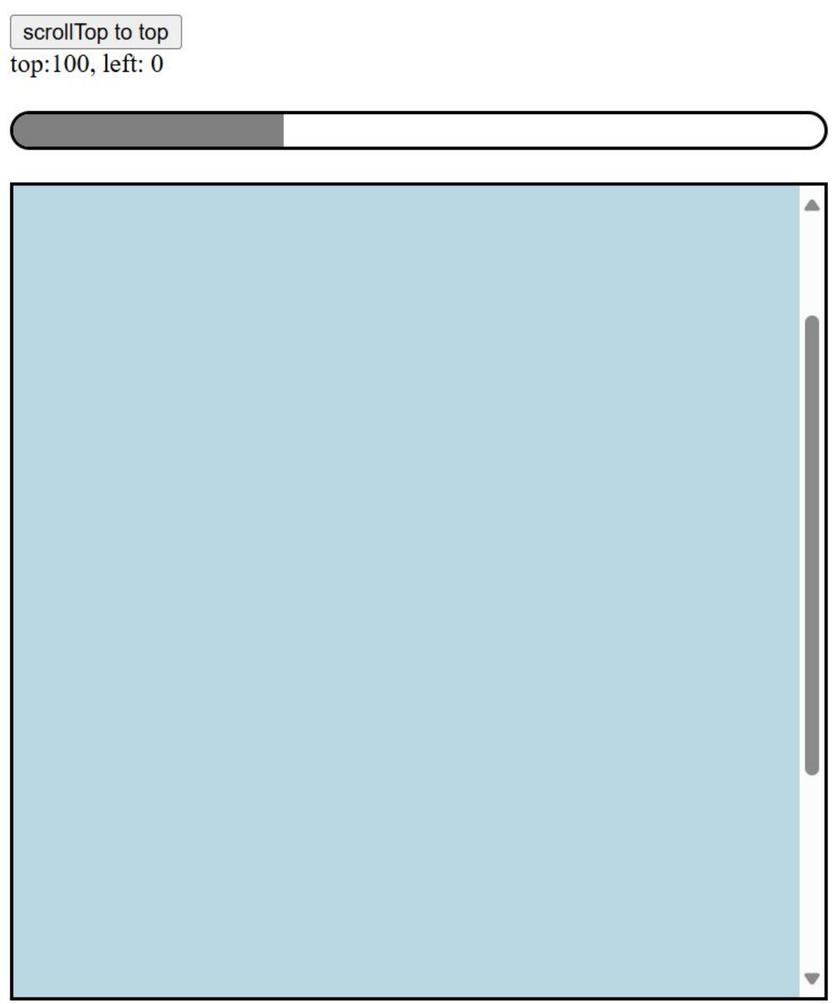
滾動事件與方法練習

1. 在畫面上建立一個容器 『 』並設定其範圍及溢出效果，子層有一個容器高度大於父層，即可呈現垂直向卷軸，並 **" 在畫面中即時顯示卷軸高度 "**。

1-1. 增加卷軸進度條。

『公式： 目前高度 / (內容高度 - 容器高度) * 100 』

1-2. 增加一個按鈕，**" 點擊之後會捲動到內容最上方 "**。



lesson 5

元素內容與屬性的處理

元素內容與屬性的處理

1. 讀取、設定元素內容
2. value屬性的讀取、設定
3. 在元素內部加入新元素
4. 在元素外部加入新元素
5. 清除、刪除元素
6. 讀取與設定元素的樣式
7. 讀取與設定元素的屬性

讀取、設定元素文字

- .text()
  - 讀取該元素與其子元素的文字內容 ⇒ $("p").text()
- .text(value)
  - 設定該元素的文字內容 ⇒ $("p").text("Hello World!!")
- .html()
  - 讀取該元素與其子元素的HTML內容 ⇒ $("p").html()
- .html(value)
  - 設定該元素的HTML內容 ⇒ $("p").html(inline element)

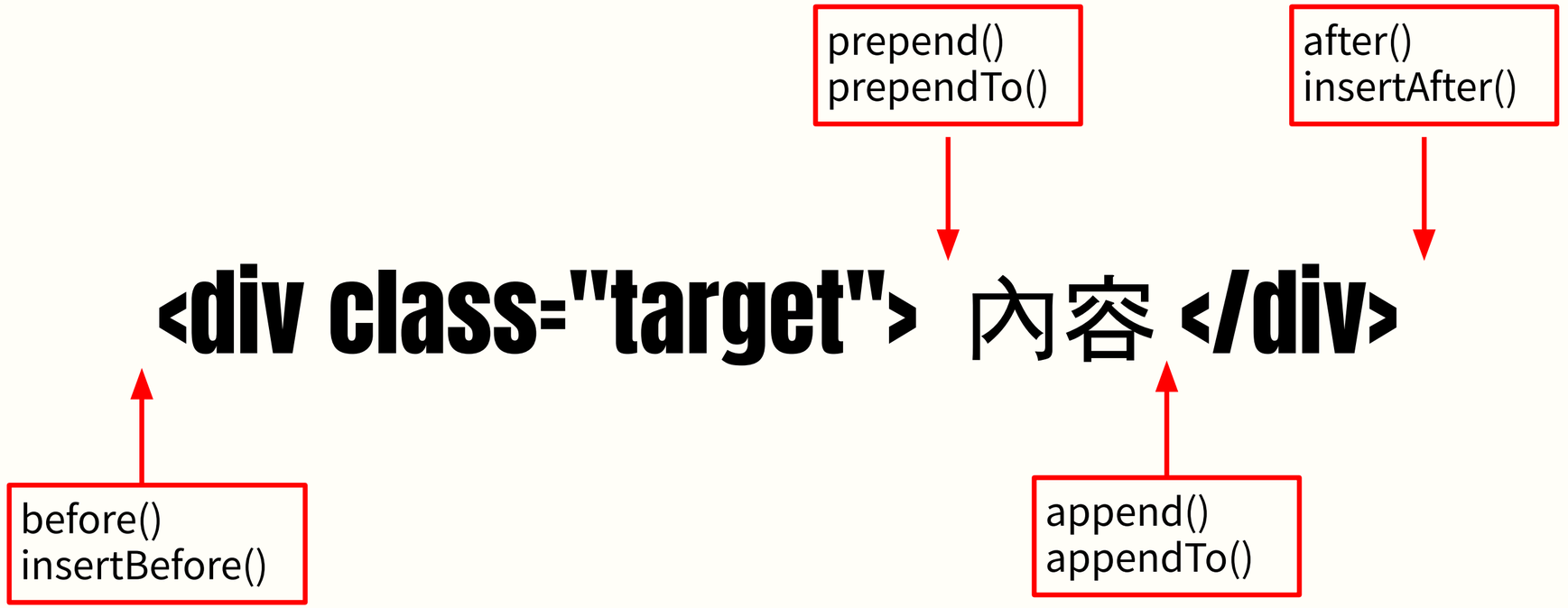
value屬性的讀取、設定

- .val()
  - 讀取該元素的value值 ⇒ $("#userName").val()
- .val(value)
  - 設定該元素的value值 ⇒ $("#userName").val("Eddie")

元素內容的練習

1. 在畫面上建立一個輸入框『 <input> 』與一個標題『 <h2> 』，在使用者輸入姓名後，將該姓名顯示在標題中" 使用者OOO "。
2. 建立一個數字的計算機，在畫面上建立兩個輸入框『<input>』、一個四則運算的下拉式選單『<select>』、一個按鈕『<input type="button">』及一個只能顯示結果輸入框『<input>』，當使用者輸入兩個數字後點擊按鈕，程式碼會自動將計算後的數字顯示在結果輸入框。

在元素內/外部加入新元素



在元素內部加入新元素

- .append(content)
  - 在該元素內插入新元素，最後一項 ⇒ $(".target").append("<h4>append text</h4>")
- .appendTo(selector)
  - 在該元素內插入新元素，最後一項 ⇒ $("<h4>appendTo text</h4>").appendTo(".target")
- .prepend(content)
  - 在該元素內插入新元素，第一項 ⇒ $(".target").prepend("<h4>prepend text</h4>")
- .prependTo(selector)
  - 在該元素內插入新元素，第一項 ⇒ $("<h4>prependTo text</h4>").prependTo(".target")

在元素外部加入新元素

- .after(content)
  - 在該元素之外插入新元素，後面 ⇒ $(".target").after("<h4>after .target</h4>")
- .insertAfter(selector)
  - 在該元素之外插入新元素，後面 ⇒ $("<h4>insertAfter .target</h4>").insertAfter(".target")
- .before(content)
  - 在該元素之外插入新元素，前面 ⇒ $(".target").before("<h4>before text</h4>")
- .insertBefore(selector)
  - 在該元素之外插入新元素，前面 ⇒ $("<h4>insertBefore text</h4>").insertBefore(".target")

清除、刪除元素

- .empty()
  - 清除該元素的內容，保留該元素 ⇒ $(".target").empty()
- .remove()
  - 直接刪除該元素 ⇒ $(".target").remove()

元素內容的練習

1. 建立一個通訊錄，使用表格『<table>』，每一個表格列有三個欄位分別是分類、姓名、電話『<select>、<input type="text">、<input type="tel">』，點擊送出按鈕『<button>』即新增一筆資料在表格最下面。

讀取與設定元素的屬性

- .attr(attributeName)
  - 讀取該元素指定的html屬性值 ⇒ $(".image").attr("src")
- .attr(attributeName, value)
  - 設定該元素html屬性值 ⇒ $(".image").attr("src","http://……")
- .removeAttr(attributeName)
  - 移除該元素的指定html屬性 ⇒ $(".image").attr("src")

讀取與設定元素的屬性

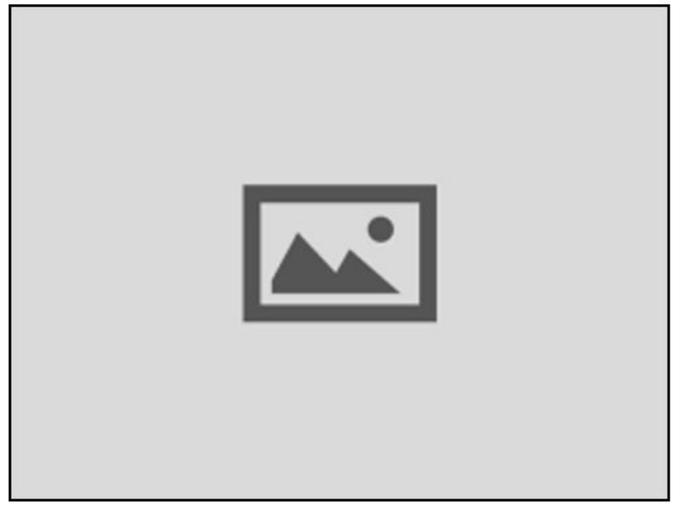
- .prop(propertyName)
  - 讀取該元素指定的屬性值 ⇒ $("#agree").prop("checked")
- .prop(propertyName, value)
  - 設定該元素html屬性值 ⇒ $("#agree").prop("checked",true)

元素屬性的練習

1. 設定一個寬度 400px / 高度 300px、2px 黑色邊框的圖片元素『 <img> 』，並建立四個按鈕分別為『 取得圖片src、取得圖片title、設定圖片一、設定圖片二 data-*="" 』，點擊圖片一、二可以設定並顯示不同圖片，點擊取得圖片屬性則會 console.log() 相關資訊。

1-1. 再增加一個按鈕『 移除屬性 』，點擊後會移除 title 屬性。

2. 在畫面上建立三個按鈕分別為『 取得prop、設定prop、送出 』，還有一個同意條款的選取方塊『 checkbox 』，當使用者勾選同意條款後，才可以點擊送出『 將按鈕的 disabled 屬性移除 』。



讀取與設定元素的樣式

- .css(propertyName)
  - 讀取該元素指定的CSS屬性值 ⇒ $(".circle").css("color")
- .css(propertyName, value)
  - 設定該元素CSS樣式 ⇒ $(".target").css("color","red")
- .css(properties)
  - 一次設定該元素多個CSS樣式 ⇒

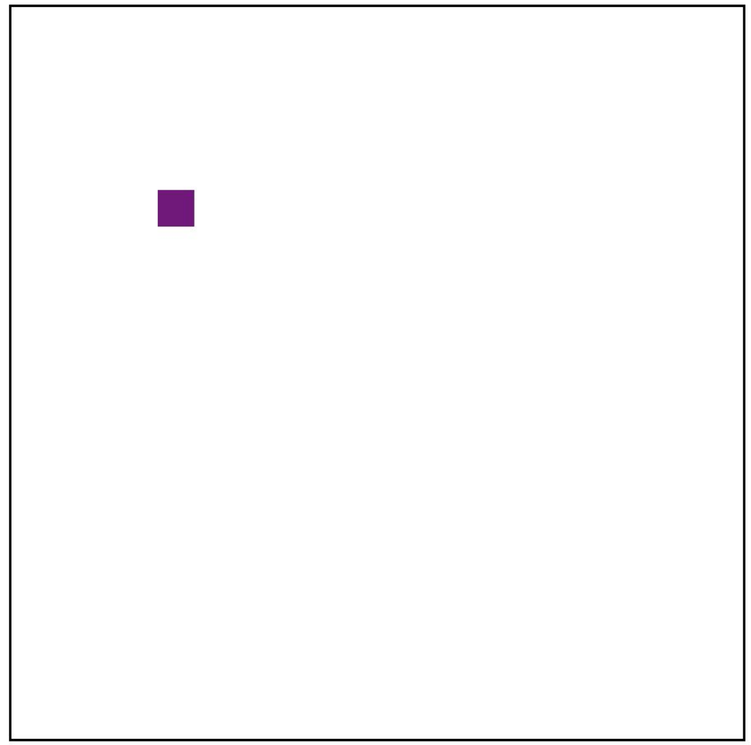
    $(".target").css({"color":"red","background-color":"blue"})

元素樣式的練習

1. 設定一個寬度/高度 600px、2px 的黑色邊框，在其中放上一個 寬度/高度 30px 的紫色正方形，透過鍵盤 左上右下『 event.whitch 分別是37、38、39、40 』讓綠色正方型可以自由移動，每次移動 30px。

1-1. 增加判斷讓紫色正方形在黑色方框中，不會超出線外。

1-2. 在移動不同方向的同時也改變正方形的背景色。



讀取與設定元素的樣式

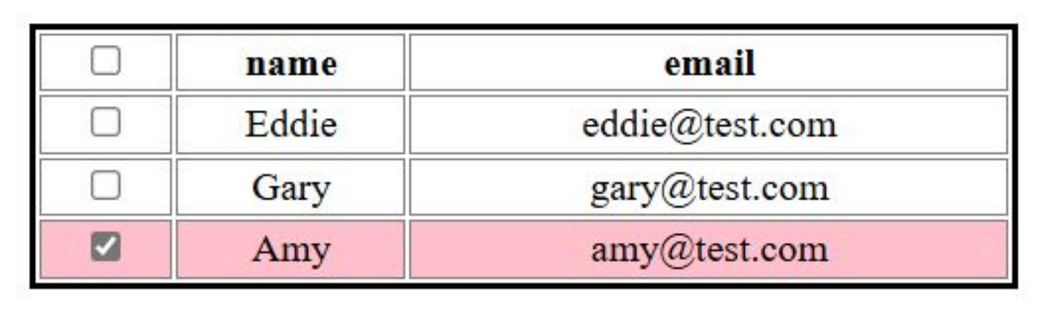
- .addClass(className)
  - 在指定元素的class屬性值中新增 className ⇒ $(".box").addClass("active")
- .removeClass(className)
  - 在指定元素的class屬性值中移除 className ⇒ $(".box").removeClass("active")
- .toggleClass(className)
  - 在指定元素的class屬性值中新增/移除 className ⇒ $(".box").toggleClass("active")
- .hasClass(className)
  - 在指定元素的class屬性值中判斷是否有 className ⇒ $(".box").hasClass("active")

元素屬性的練習

1. 在畫面上建立一個通訊錄，使用表格元素『 <table> 』搭配選取方塊『 <input type="checkbox"> 』，當表格中的每一項資料被勾選時，其背景色變成粉紅色『 addClass()、removeClass() 』。

1-1. 當全選項目被勾選或取消勾選時，可以改變下方的每一項選取方塊的狀態『 prop("checked") 』。

1-2. 當下方的每一項都被勾選時，也可以連動全選方塊。『 計算有被勾選之數量 』



lesson 6

jQuery Animate

jQuery Animate

1. 元素顯示與隱藏 hide()、show()
2. 元素淡入與淡出 fadeOut()、fadeIn()
3. 元素滑入與滑出 slideUp()、slideDown()
4. 元素顯示與隱藏 toggle()
5. fade、slide 搭配 toggle()
6. 自訂動畫 animate()
7. 第三方動畫函式庫 animate css

元素顯示與隱藏 - hide()

- .hide()
  - 將指定元素隱藏在畫面中，增加display: none ⇒ $(".box").hide()
- .hide(duration, [complete])
  - [duration] ⇒ 設定動畫時長，預設400毫秒
  - [complete] ⇒ 當動畫執行完畢時執行的函數（funtion）
    - 範例： $(".box").hide(900,function(){ // 要執行的程式 })
- .hide(options)
  - options ⇒ duration、easing、queue、specialEasing、step、progress、complete、start、done、fail、always
- .hide([duration], [easing], [complete])

元素顯示與隱藏 - show()

- .show()
  - 將已隱藏的元素顯示在畫面中，移除display: none ⇒ $(".box").show()
- .show(duration, [complete])
  - [duration] ⇒ 設定動畫時長，預設400毫秒
  - [complete] ⇒ 當動畫執行完畢時執行的函數（funtion）
    - 範例： $(".box").show(900,function(){ // 要執行的程式 })
- .show(options)
  - options ⇒ duration、easing、queue、specialEasing、step、progress、complete、start、done、fail、always
- .show([duration], [easing], [complete])

元素顯示與隱藏 - fadeOut()

- .fadeOut()
  - 將指定元素隱藏在畫面中，先執行opacity: 1到0後增加display: none ⇒ $(".box").fadeOut()
- .fadeOut(duration, [complete])
  - [duration] ⇒ 設定動畫時長，預設400毫秒
  - [complete] ⇒ 當動畫執行完畢時執行的函數（funtion）
    - 範例： $(".box").fadeOut(900,function(){ // 要執行的程式 })
- .fadeOut(options)
  - options ⇒ duration、easing、queue、specialEasing、step、progress、complete、start、done、fail、always
- .fadeOut([duration], [easing], [complete])

元素顯示與隱藏 - fadeIn()

- .fadeIn()
  - 將已隱藏的元素顯示在畫面中，先移除display: none後執行opacity: 0到1 ⇒ $(".box").fadeIn()
- .fadeIn(duration, [complete])
  - [duration] ⇒ 設定動畫時長，預設400毫秒
  - [complete] ⇒ 當動畫執行完畢時執行的函數（funtion）
    - 範例： $(".box").fadeIn(900,function(){ // 要執行的程式 })
- .fadeIn(options)
  - options ⇒ duration、easing、queue、specialEasing、step、progress、complete、start、done、fail、always
- .fadeIn([duration], [easing], [complete])

元素顯示與隱藏 - slideUp()

- .slideUp()
  - 將指定元素隱藏起來，先調整height、overflow後執行display: none ⇒ $(".box").slideUp()
- .slideUp(duration, [complete])
  - [duration] ⇒ 設定動畫時長，預設400毫秒
  - [complete] ⇒ 當動畫執行完畢時執行的函數（funtion）
    - 範例： $(".box").slideUp(900,function(){ // 要執行的程式 })
- .slideUp(options)
  - options ⇒ duration、easing、queue、specialEasing、step、progress、complete、start、done、fail、always
- .slideUp([duration], [easing], [complete])

元素顯示與隱藏 - slideDown()

- .slideDown()
  - 將已隱藏的元素顯示在畫面中，先移除display: none後調整height、overflow ⇒ $(".box").slideDown()
- .slideDown(duration, [complete])
  - [duration] ⇒ 設定動畫時長，預設400毫秒
  - [complete] ⇒ 當動畫執行完畢時執行的函數（funtion）
    - 範例： $(".box").slideDown(900,function(){ // 要執行的程式 })
- .slideDown(options)
  - options ⇒ duration、easing、queue、specialEasing、step、progress、complete、start、done、fail、always
- .slideDown([duration], [easing], [complete])

元素顯示與隱藏 - toggle()

- .toggle()
  - 切換顯示/隱藏狀態 ⇒ $(".box").toggle()
- .toggle(duration, [complete])
  - [duration] ⇒ 設定動畫時長，預設400毫秒
  - [complete] ⇒ 當動畫執行完畢時執行的函數（funtion）
    - 範例： $(".box").toggle(900,function(){ // 要執行的程式 })
- .toggle(options)
  - options ⇒ duration、easing、queue、specialEasing、step、progress、complete、start、done、fail、always
- .toggle([duration], [easing], [complete])

fade、slide 搭配 toggle()

- .fadeToggle()
- .slideToggle()

自訂動畫 animate()

- .animate(properties, [duration], [easing], [complete])
  - [properties] ⇒ 設定的css樣式，表示方式 { CSS屬性: "CSS屬性值" }。
  - [duration] ⇒ 動畫執行時間長度。
  - [easing] ⇒ 動滑執行的效果，分為 swing、 linear、自定義。
  - [complete] ⇒ 動畫完成後執行的函數（funciton）。
    - 範例：
    - $(".box").animate({

      left: "+=50",

      borderRadius: "+=10"

      },1000)

自訂動畫 animate()

- .animate()
  - 串接不同的animate。
    - 範例：
    - $(".box").animate({
      left: "+=50",
      borderRadius: "+=10"
      },1000).animate({
      top: "+=50",
      borderRadius: "+=10"
      },1000)

animate css

- 第三方動畫函式庫：https://animate.style/
- 使用方式：
  - 1. 使用<link>元素引入CDN。
    - <link rel="stylesheet"
      href="https://cdnjs.cloudflare.com/ajax/libs/animate.css/4.1.1/animate.min.css" />
  - 2. 右側選單選擇喜歡的樣式，並複製class屬性值，於指定元素身上加入class。
    -
    - 紅色：固定要加入的屬性值
    - 藍色：右側選單複製的樣式

animate css

- 搭配已寫好的 JavaScript 『 function animateCSS() 』，來動態執行效果。
  - 範例：
  - <button id="zoomOut">zoomOut</button>
  - $("#zoomOut").click(function () {
    animateCSS(".box", "zoomOut").then(function () {
    $(".box").css("display", "none");
    });
    });

animista

- 第三方動畫函式庫：https://animista.net/

紅色：選擇效果
藍色：客製化設定
綠色：複製CSS樣式

jQuery UI

- 第三方動畫函式庫：https://jqueryui.com/
- 使用方式：
  - 1. 下載Stable版本，並放入專案。
    - 透過link元素引入檔案中的 jquery-ui.css。
    - <link rel="stylesheet" href="./jquery-ui-1.14.1/jquery-ui.css" />
    - 在body元素下方要先引入jquery-3.7.1.js後再引入jquery-ui.js。
    - <script src="./jquery-3.7.1.js"></script>
    - <script src="./jquery-ui-1.14.1/jquery-ui.js"></script>
  - 2. 找到需要的功能，引入即可。
    - Widgets > datepicker

lesson 7

ajax

ajax Shorthand Methods

- .get(url, function)
- .post(url, function)
- .getJSON(url, function)
  - [url] ⇒ 表示伺服器端的某一支程式
  - [function] ⇒ 請求的回應成功時會呼叫的函式。
  - 範例：
  - $.getJSON("url", function(data) {
  - console.log("response data", data);
  - });

ajax

- .ajax(url, [settings])
  - settings：
    - [url] ⇒ 表示伺服器端的某一支程式。
    - [type] ⇒ get 或 post。
    - [data] ⇒ 傳遞到伺服器端的資料（字串、物件、陣列）。
    - [dataType] ⇒ 伺服器端回傳的資料格式。（ xml, json, script, text, html ）
    - [success] ⇒ 請求的回應成功時會呼叫的函式。
    - accepts、async、beforeSend、cache、complete、contents、contentType、converts、crossDomain。

附錄

其他工具

其他工具

1. 表格 - DataTables https://datatables.net/
2. 提視窗 - SweetAlert2 https://sweetalert2.github.io/
3. 瀑布流 - Masonary https://masonry.desandro.com/
4. 圖片燈箱 - lightbox2 https://lokeshdhakar.com/projects/lightbox2/
5. 視覺特效 - wow.js - https://wowjs.uk/
6. 圖片輪播 - swiper https://swiperjs.com/

附錄

jQuery 其他 API

jQuery 其他 API

1. .each()
2. .inArray()
3. .grep()
4. .map()
5. .data()

jQuery 其他 API

- .each(function)
  - 將包裝在jQuery物件中的元素，一項一項單獨讀出處理。
  - $(selector).each(function(index, element){ console.log(index, value) })
- .each(array, function)
  - let arrayOne = ["Eddie", "iSpan", "JavaScript", "jQuery"]
  - $.each(arrayOne ,function(index, value){ console.log(index, value) })
- .each(object, function)
  - let objectOne = { name: "Eddie", tel: "02-6631-6588",email: "eddie@test.com", }
  - $.each(objectOne ,function(key, value){ console.log(key, value) })

jQuery 其他 API

- .inArray(value, array, [fromIndex])
  - 在陣列中查找指定值，並返回其索引值。如果未找到該值，則返回 -1
  - [value] ⇒ 要查找的值。
  - [array] ⇒ 目標陣列。
  - [fromeIndex] ⇒ 從指定第幾項開始搜尋。
  - 範例：
    - let fruits = ["apple", "banana", "guava", "orange"]
    - let isInArray = $.inArray("orange", fruits)

jQuery 其他 API

- .grep(array, function, [invert])
  - 用於過濾陣列中的資料。
  - [array] ⇒ 目標陣列。
  - [function] ⇒ 將陣列中的資料逐筆讀取 (elementOfArray , indexOfArray)。
  - [invert] ⇒ 預設為false，將回傳符合的項目，設定為true則返回不符合的項目。

jQuery 其他 API

- .grep()
  - 範例：
    - let products = [
    - { name: "iPhone 12 mini", category: "phone" },
    - { name: "iPhone 14", category: "phone" },
    - { name: "iPad mini", category: "pad" },
    - { name: "iPad pro", category: "pad" },
    -]
    - let newProduct = $.grep(products, function(product,index){
    - return product.category == "pad"
    - })
    - console.log("newProduct", newProduct);

jQuery 其他 API

- .map(array, function)
  - 將陣列中的資料逐一處理，並返回新的陣列。
  - [array] ⇒ 目標陣列。
  - [function] ⇒ 將陣列中的資料逐筆讀取處理 (elementOfArray , indexOfArray)。
  - 範例：
    - const num = [99, 50, 77, 88, 99, 656];
    - let newNum = $.map(num, function (num, index) {
    - return num * index;
    - });
    - console.log(newNum);

jQuery 其他 API

- .data(key)
  - 取得指定元素中 data-* 屬性的值。
  - [key] ⇒ 屬性名稱。
- .data(key, value)
  - 透過 data-* 屬性的方式，儲存一些資料在元素中。
  - [key] ⇒ 屬性名稱。
  - [value] ⇒ 屬性值。
    - 範例：
    - <button id="btn" data-name="eddie" data-age="500" >get data</button>
    - $("#btn").on("click",function(){ console.log($(this).data("name")) })