

下載範例檔案

- https://github.com/ck1001099/cpp_course_2019winter
- extra -> Lecture -> Object_template.cpp

C++程式設計基礎 extra

陳毅

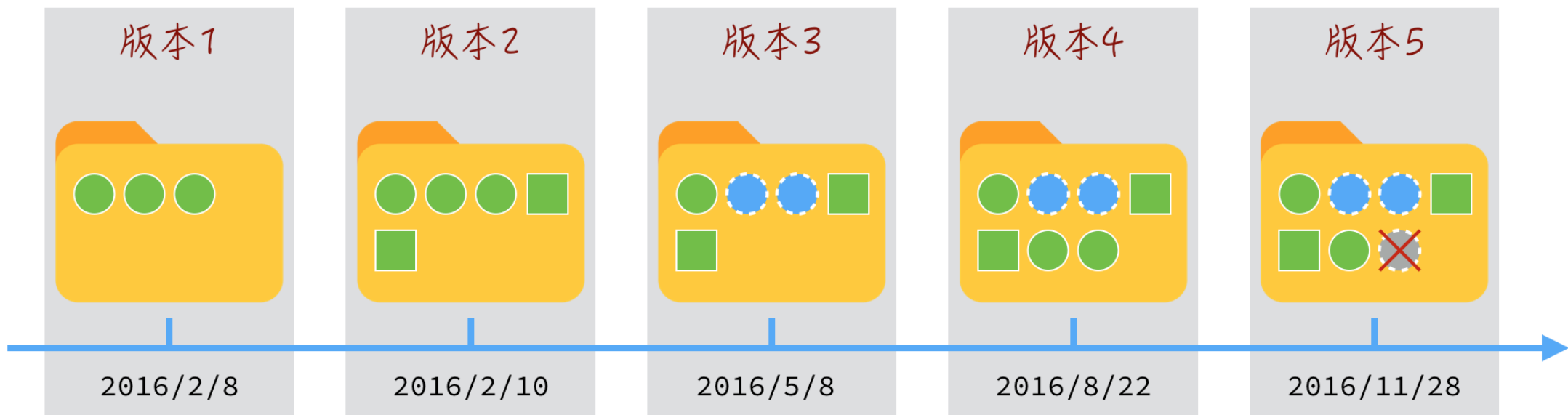
概要

- 專案管理：Sourcetree + GitHub
- 物件導向補充 – 靜態成員(static member)
- 多載函數（Ch12）
- 進階演算法
 - 更多的排序演算法：Insertion sort、Quick sort
 - Dynamic programming

專案管理：Sourcetree + GitHub

Git是什麼？

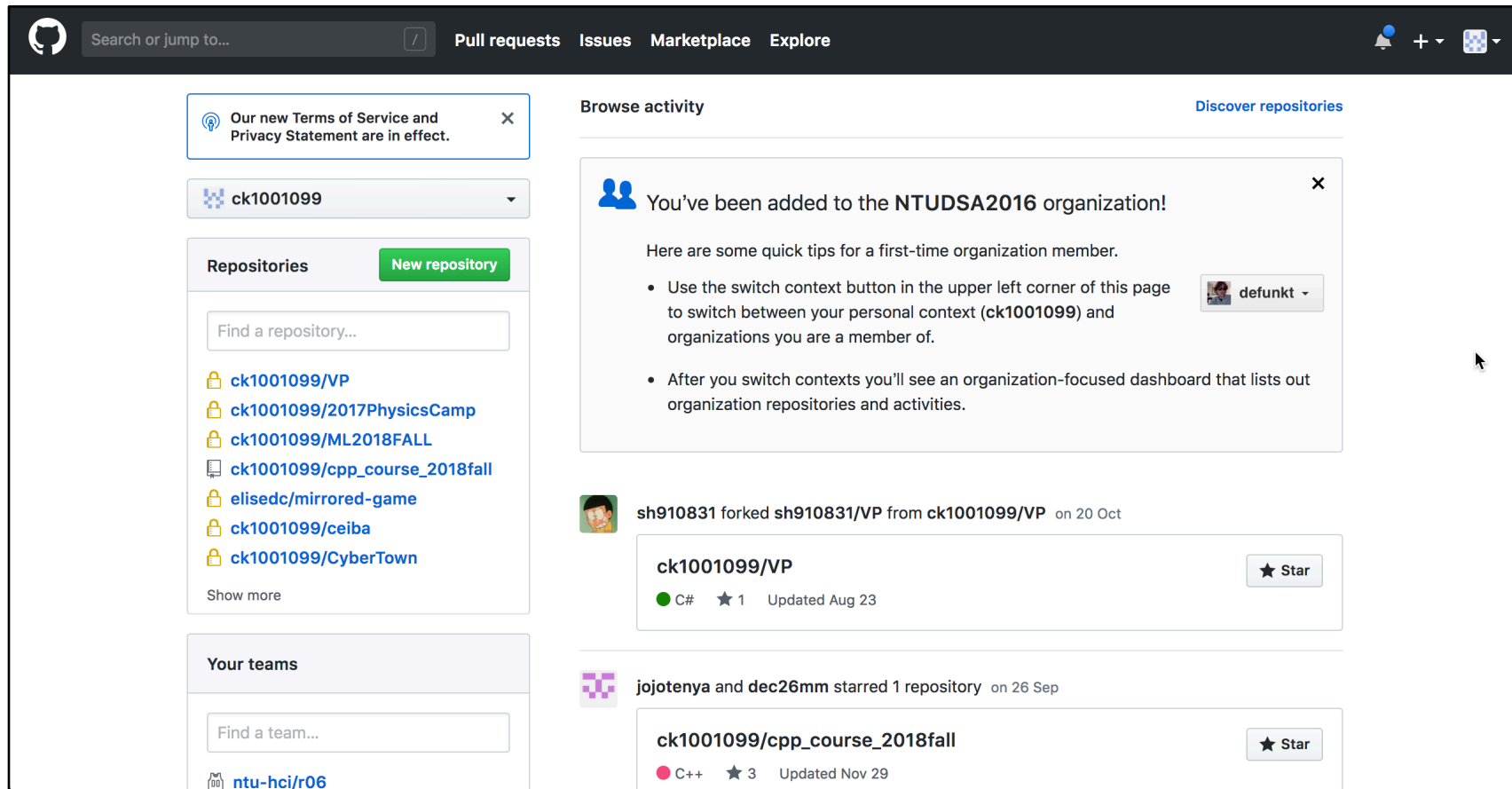
- Git是一種分散式版本的**版本控制系統(Version Control System)**。



- 常見的線上版本控制平台：GitHub、BitBucket等。


GitHub 介紹

- GitHub: <https://github.com/>



Sourcetree 介紹

- Sourcetree: <https://www.sourcetreeapp.com/>

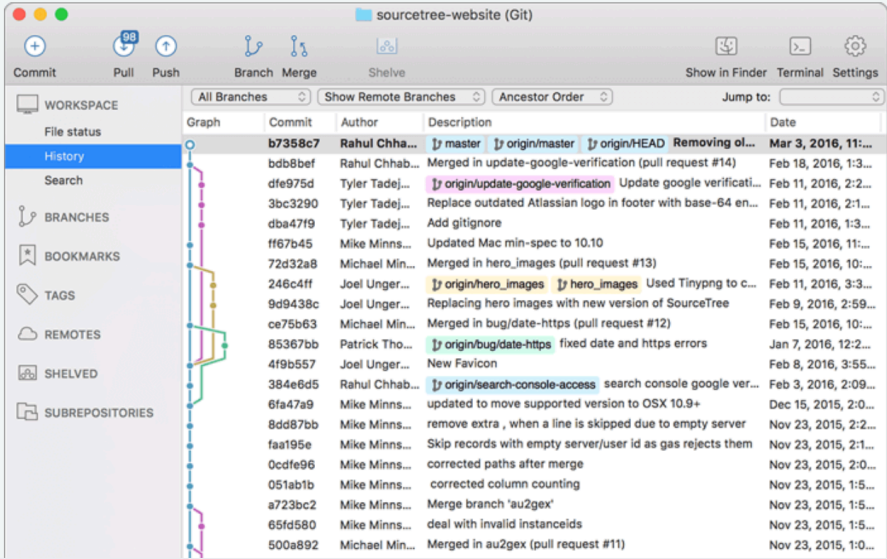
 Sourcetree

Download free

Simplicity and power in a beautiful Git GUI

Download for Mac OS X

Also available for Windows

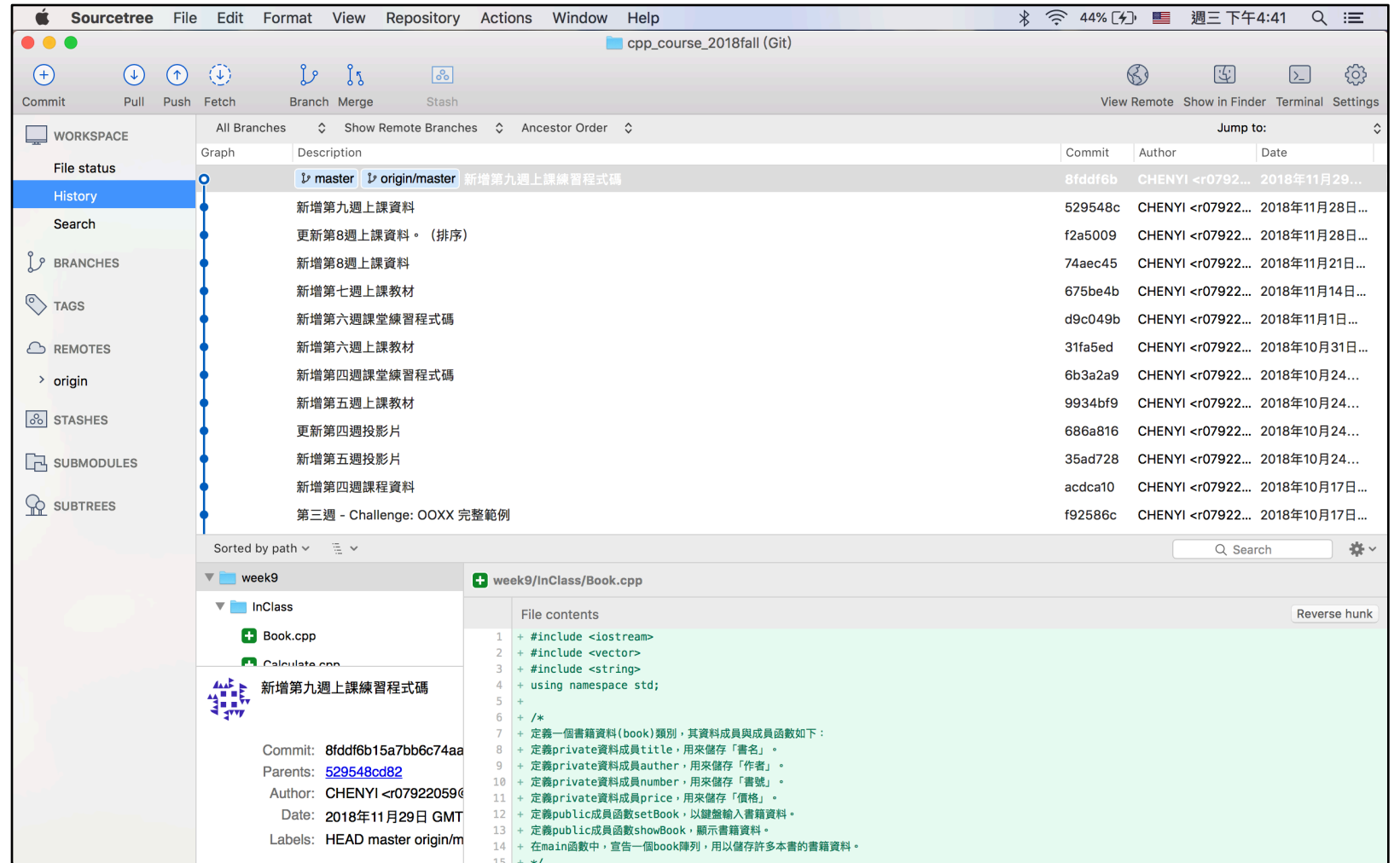
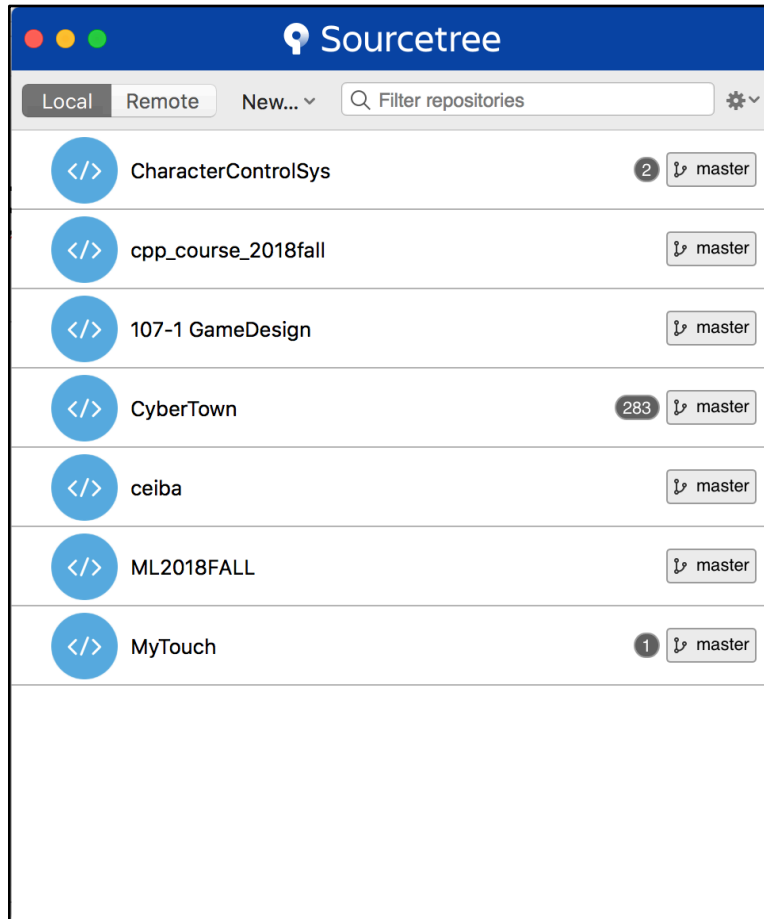


Graph	Commit	Author	Description	Date
	b7358c7	Rahul Chha...	origin/master origin/HEAD Removing ol...	Mar 3, 2016, 11:...
	bdb8bef	Rahul Chhab...	Merged in update-google-verification (pull request #14)	Feb 18, 2016, 1:3...
	dfe975d	Tyler Tadej...	origin/update-google-verification Update google verificati...	Feb 11, 2016, 2:2...
	3bc3290	Tyler Tadej...	Replace outdated Atlassian logo in footer with base-64 en...	Feb 11, 2016, 2:1...
	dba47f9	Tyler Tadej...	Add gitignore	Feb 11, 2016, 1:3...
	ff67b45	Mike Minns...	Updated Mac min-spec to 10.10	Feb 15, 2016, 11:...
	72d32a8	Michael Min...	Merged in hero_images (pull request #13)	Feb 15, 2016, 10:...
	246c4ff	Joel Unger...	origin/hero_images hero_images Used TinyPNG to c...	Feb 11, 2016, 3:3...
	9d9438c	Joel Unger...	Replacing hero images with new version of SourceTree	Feb 9, 2016, 2:59...
	ce75b63	Michael Min...	Merged in bug/date-https (pull request #12)	Feb 15, 2016, 10:...
	85367bb	Patrick Tho...	origin/bug/date-https fixed date and https errors	Jan 7, 2016, 12:2...
	4f9b557	Joel Unger...	New Favicon	Feb 8, 2016, 3:55...
	384e6d5	Rahul Chhab...	origin/search-console-access search console google ver...	Feb 3, 2016, 2:09...
	6fa47a9	Mike Minns...	updated to move supported version to OSX 10.9+	Dec 15, 2015, 2:0...
	8dd87bb	Mike Minns...	remove extra , when a line is skipped due to empty server	Nov 23, 2015, 2:2...
	faa195e	Mike Minns...	Skip records with empty server/user id as gas rejects them	Nov 23, 2015, 2:1...
	0cdf96	Mike Minns...	corrected paths after merge	Nov 23, 2015, 2:0...
	051ab1b	Mike Minns...	corrected column counting	Nov 23, 2015, 1:5...
	a723bc2	Mike Minns...	Merge branch 'au2gex'	Nov 23, 2015, 1:5...
	65fd580	Mike Minns...	deal with invalid instanceids	Nov 23, 2015, 1:5...
	500a892	Michael Min...	Merged in au2gex (pull request #11)	Nov 23, 2015, 1:0...

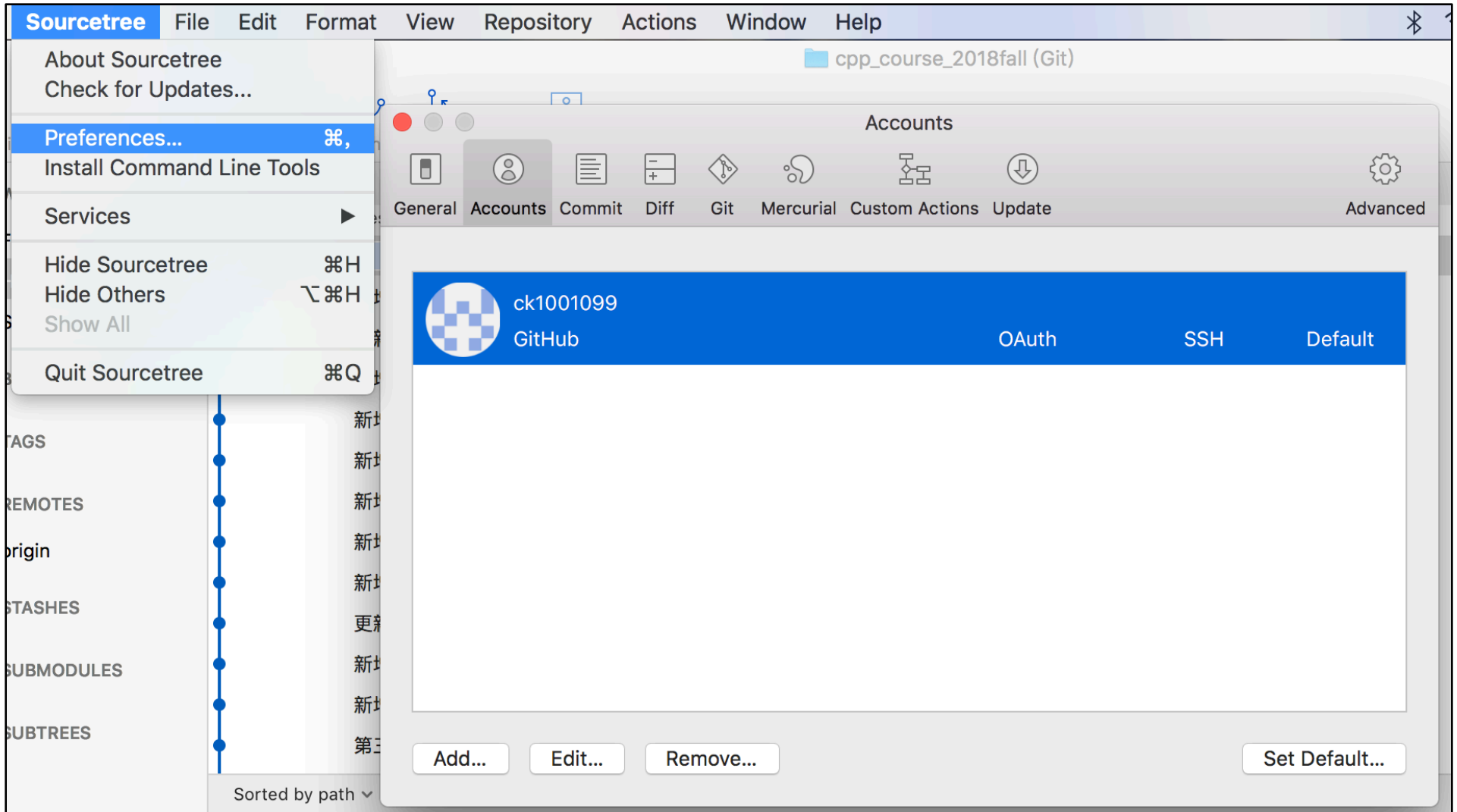
A free Git client for Windows and Mac

Sourcetree simplifies how you interact with your Git repositories so you can focus on coding. Visualize and

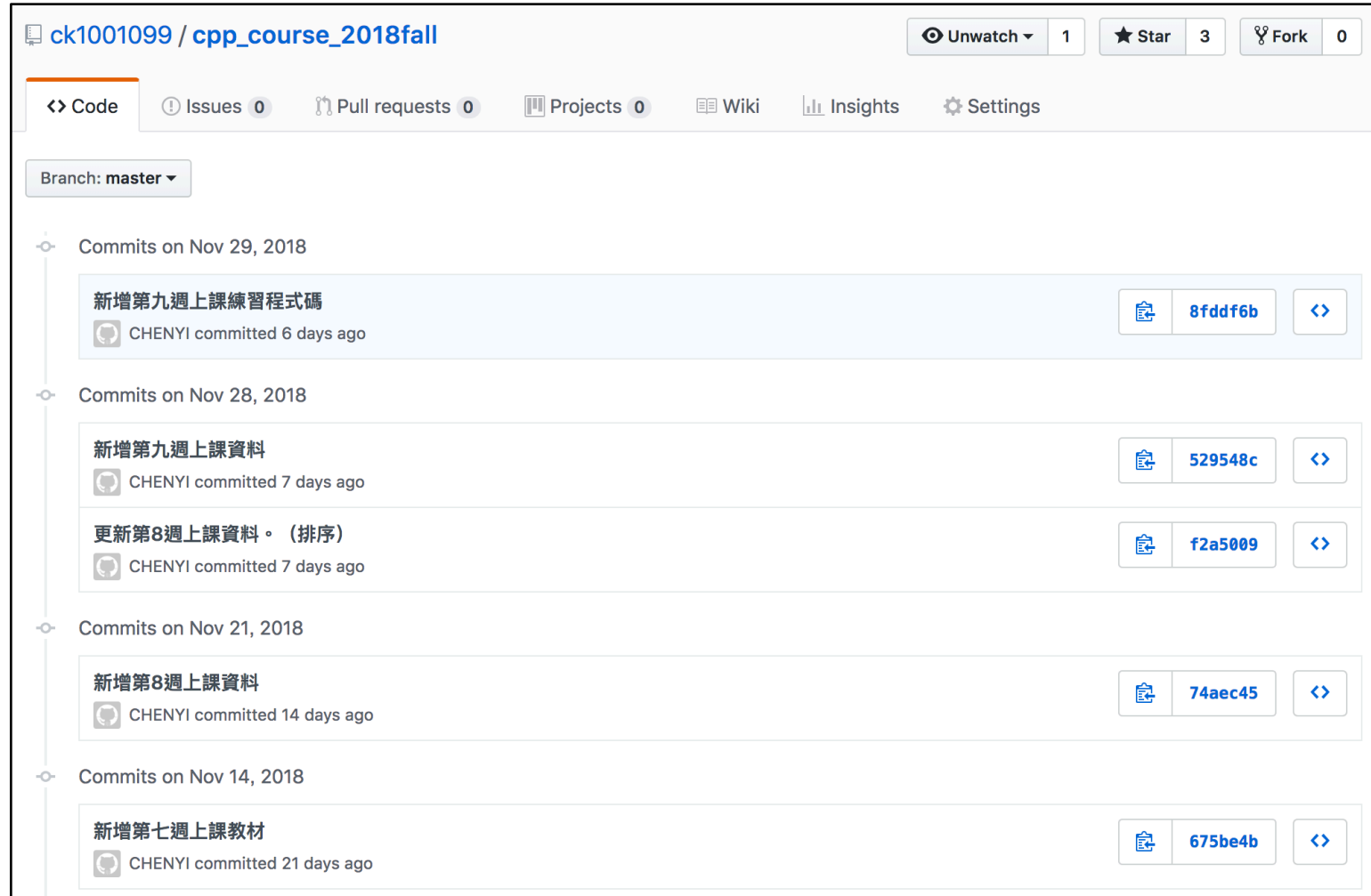
Sourcetree + GitHub



Sourcetree + GitHub



Sourcetree + GitHub



補充資料

- 30 天精通 Git 版本控管
<https://github.com/doggy8088/Learn-Git-in-30-days>
- GitHub版本控制
<https://hackmd.io/s/Bk2AaU6o>
- Git的基本介紹
https://backlog.com/git-tutorial/tw/intro/intro1_1.html
- 一步一步教你用 GitHub + SourceTree 做版本控制
<https://hackmd.io/s/Bymbf3y2>

物件導向補充

靜態成員(static member)

- 若某個資料成員被宣告為static，則其他同類別的物件皆可共享該靜態成員的資料。
- 若某個成員函數被宣告為static，則該static成員函數，不可呼叫非static的成員函數。
- 使用方法：宣告的型態前，加上「static」。

```
static 資料型態 變數名稱;  
static 函數型態 函數名稱(參數列){ 敘述區; }
```

靜態成員(static member)

- 靜態的資料成員，必須要進行初始化，而且也只能被初始化一次。

```
class Test{  
public:  
    Test(int k){ this->k = k; }  
    void Add1(){ number = number + 1; }  
    int Get(){ return number+k; }  
private:  
    static int number;  
    int k;  
};  
  
int Test::number = 0;
```

練習

- 定義一個物體(Object)類別，其資料成員與成員函數如下：
 - 定義一個private資料成員name，儲存該物體名稱。
 - 定義一個private資料成員index，儲存該物體編號。
 - 定義一個static private資料成員count，記錄已經建立多少個物體。
 - 定義一個static private資料成員objList，用於儲存被建立的物體的指標。
 - 定義建立者(constructor)函數，初始化name，將count值加1，並將指標存入objList。
 - 定義破壞者(destructor)函數，將count值減1，並根據編號從objList中移除指標。
 - 定義一個static public成員函數info，輸出總物體數目以及各物體編號及名稱。

多載函數

什麼是多載(overloading)？

- 四則運算也是函數的一種。
- $3+5$ ，實際上就是 *int operator+ (3, 5)*
- $6.3-1.8$ ，實際上就是 *double operator- (6.3, 1.8)*

什麼是多載(overloading)？

- 關於加法，加數與被加數有很多種組合。
- 整數 + 整數： $3+5$, $62+10$, ...
- 整數 + 小數： $7+1.3$, $6+3.14$, ...
- 小數 + 整數： $5.32+13$, $872.24+524$, ...
- 小數 + 小數： $34.43+235.73$, $1.548+745.231$, ...
- 那加法的函數雛形到底長什麼樣子呢？

什麼是多載(overloading)？

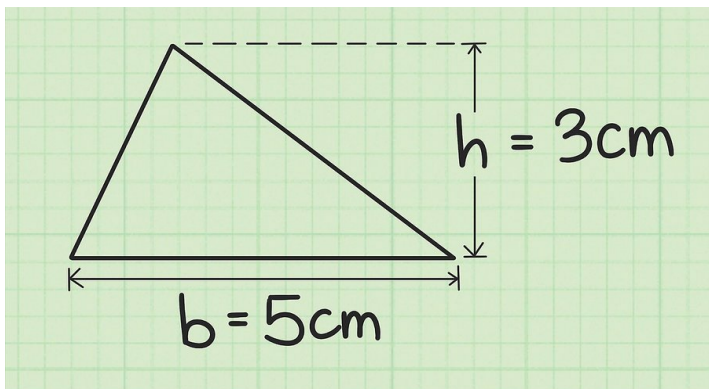
- Ans：加法的函數雛形其實有很多個！
- 整數 + 整數：int operator+ (int, int)
- 整數 + 小數：double operator+ (int, double)
- 小數 + 整數：double operator+ (double, int)
- 小數 + 小數：double operator+ (double, double)
- 同個函數名稱卻有不同的定義及功能，這就稱作多載(**overloading**)。

練習

- 想要計算一個三角形的面積有很多不同的算法，例如：

$$A = \sqrt{s(s-a)(s-b)(s-c)}$$

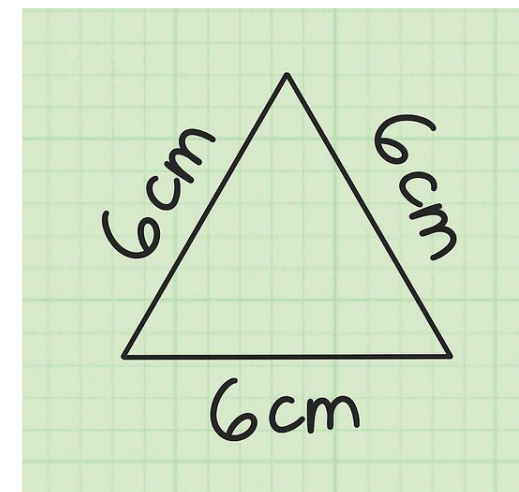
```
double area(int, int, int);
```



```
double area(int, int);
```

$$\text{Area} = \frac{bc}{2} \sin A$$

```
double area(int, int, double);
```



```
double area(int);
```

多載函數還能做到的事

- 若你定義了一個類別用來儲存複數，那要怎麼做複數四則運算呢？
 - 可自己定義屬於自己的四則運算(+, -, *, /)。

```
class Test{
public:
    Test(){ a = b = 0; }
    Test(int n, int m){ a = n; b = m; }
    Test operator+(Test);
    void display(){ cout << a << " " << b << endl; }
private:
    int a, b;
};

Test Test::operator+(Test obj){
    Test tmp;
    tmp.a = a + obj.a;
    tmp.b = b + obj.b;
    return tmp;
}
```

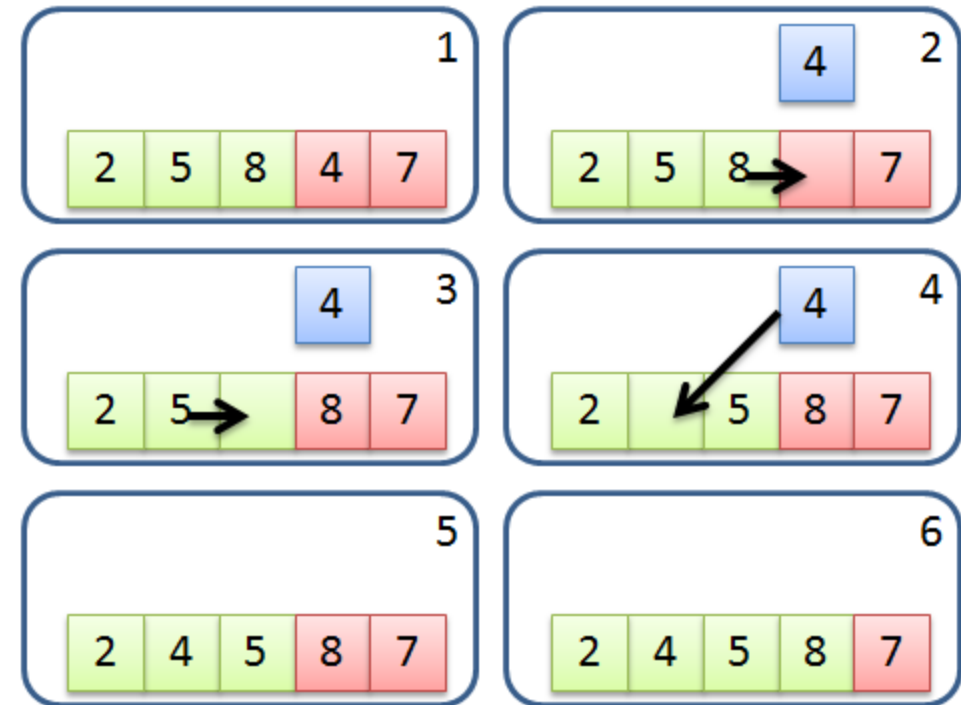
進階演算法

插入排序法(Insertion sort)

- **插入排序法(Insertion Sort)**是排序演算法的一種，也是一種簡單容易理解的演算法。
- 對於未排序資料，在已排序序列中從後向前掃描，找到相應位置並插入。
- 在從後向前掃描過程中，需要反覆把已排序元素逐步向後挪位，為最新元素提供插入空間。

插入排序法(Insertion sort) – 實作

```
void insertion_sort(int arr[],int len){  
    for(int i=1;i<len;i++){  
        int key=arr[i];  
        int j=i-1;  
        while((j>=0) && (key<arr[j])){  
            arr[j+1]=arr[j];  
            j--;  
        }  
        arr[j+1]=key;  
    }  
}
```



插入排序法(Insertion sort) – 複雜度分析

- 最佳時間複雜度： $O(n)$
- 平均時間複雜度： $O(n^2)$
- 最差時間複雜度： $O(n^2)$
- 空間複雜度： $O(1)$

快速排序法(Quick sort)

- **快速排序法(Quick sort)**是排序演算法的一種，使用***Divide and Conquer***的演算法來實作。
- 其概念是從數列中挑選一個基準點，大於基準的放一邊，小於的放一邊，如此循環最後可完成排序。
- 依照以下步驟進行(遞增為例)：
 1. 數列中選擇一元素作為基準點(pivot)。
 2. 小於此元素的移至基準的左邊，大於此元素的移至右邊，相等的任意放。
 3. 基準點左邊和右邊視為兩個數列，並重複做以上動作直到數列剩下一個或零個元素。

快速排序法(Quick sort) – 實作

```
function sort(list)
  if list.length < 2
    return list
  end if
  pivot = 從list取出一基準點
  var less, greater, result
  for i = 0; i < list.length; ++i
    if list[i] > pivot
      greater.add(list[i])
    else
      less.add(list[i])
    end if
  end for
  result.add(sort(less))
  result.add(pivot)
  result.add(sort(greater))
  return result;
end function
```



快速排序法(Quick sort) – 複雜度分析

- 最佳時間複雜度： $O(n \log n)$
- 平均時間複雜度： $O(n \log n)$
- 最差時間複雜度： $O(n^2)$
- 空間複雜度： $O(n)$

Dynamic Programming (DP)

範例：階乘（Factorial）

$1 \times 2 \times 3 \times \dots \times N$ 。整數 1 到 N 的連乘積。 N 階乘。 $N!$ 。

$N!$ 源自 $(N-1)!$ ，如此就遞迴分割問題了。

$$\underbrace{1 \times 2 \times 3 \times \dots \times (N-2) \times (N-1)}_{(N-1)!} \times N$$

陣列的每一格對應每一個問題。設定第一格的答案，再以迴圈依序計算其餘答案。

0!	1!	2!	3!	4!	5!	6!	7!	8!	9!	10!
	1									
	1	2								
	1	2	6							

參考來源：<http://www.csie.ntnu.edu.tw/~u91029/DynamicProgramming.html>

更多演算法

- 演算法筆記：<http://www.csie.ntnu.edu.tw/~u91029/index.html>
- LeetCode – Algorithm: <https://leetcode.com/problemset/algorithms/>

作業

- Reading: 課本 Ch11.5.3, Ch12
- 若遇到程式相關問題，歡迎隨時寄信至：r07922059@ntu.edu.tw