

2017 1 학기 깃허브 개요

자기주도적 학습

깃허브 처음 활용

내용 목차

1	처음으로 깃허브에 파일 올리기.....	2
	깃허브 계정 생성과 로그인.....	2
	깃허브 저장소 생성과 파일 업로드	18
	데스크 탑의 폴더와 파일 구성과 깃허브에 업로드	오류! 책갈피가 정의되어 있지 않습니다.

1 깃허브 계정 생성과 로그인

깃허브란?

깃허브는 분산 버전관리 도구인 깃(Git)을 지원하는 웹 호스팅 서비스이다. 깃을 지원하는 웹 호스팅 서비스로는 깃허브(GitHub)와 깃랩(GitLab)이 대표적으로, 깃허브는 개인이 공개로 웹 서비스로 활용하면 무료이며, 깃랩은 소수의 팀원이 프로젝트를 수행하는 웹 서비스로 활용해도 무료이다. 깃허브는 대표적인 오픈 소스 프로젝트 저장소로도 널리 활용되고 있는데, 오픈 소스 프로젝트 저장소의 대표적인 사이트로는 소스포지(SourceForge)와 구글 코드(ZGoogle Code) 등이 있다.

오픈 소스란 소프트웨어 등을 만들 때 해당 소프트웨어의 소스코드를 무료 공개, 배포하는 것으로 리눅스(Linux) 운영체제가 대표적이다. 오픈 소스 소프트웨어는 누구나 무료로 이용할 수 있으며, 공개된 코드를 기반으로 프로그램을 마음대로 변형할 수도 있다. 인터넷을 이용하는 다수의 기술자가 소프트웨어를 공동으로 개발할 경우 보다 나은 소프트웨어를 단기간에 개발할 수 있다는 개념에서 오픈 소스가 추진되었다.

깃은 2006 년경 비트키퍼(BitKeeper)라는 리눅스 커널 개발에 쓰던 분산형 패치 도구에 대한 대안으로 리누스 토발즈가 직접 개발한 분산형 소스 관리 시스템(Distributed Source Control Management)이다.

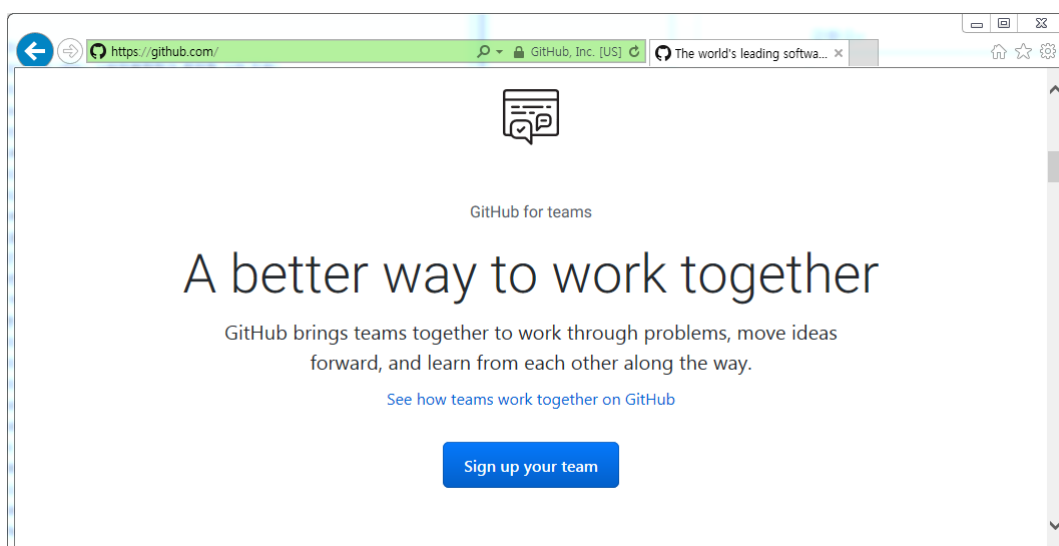
2 깃허브 계정 생성과 로그인

깃허브 계정을 만들어 보자

깃허브 계정 생성

처음으로 깃허브 사이트에 접속하여 계정을 만들도록 하자. 홈페이지의 중간 부분의 문구를 보면 깃허브는 팀을 위한 사이트로 “함께 일하는 좋은 방법”을 추구하는 서비스인 것을 알 수 있다.

- www.github.com



Tips:

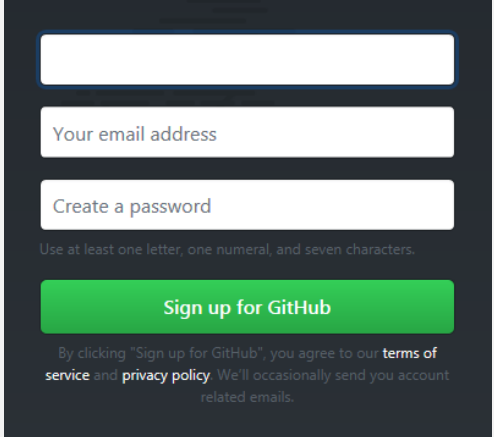
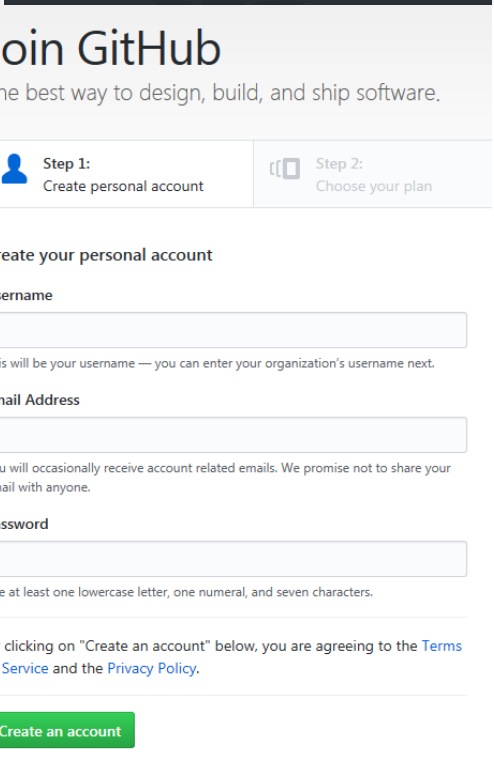
홈페이지의 문구를 살펴보면 다음과 같다.

“GitHub 은 여러분의 작업 방식에서 영감을 얻은 개발 플랫폼입니다. 수백만 명의 다른 개발자와 함께 코드를 관리하고, 프로젝트를 관리하며, 소프트웨어를 구축하십시오.”

GitHub is a development platform inspired by the way you work. Host code, manage projects, and build software alongside millions of other developers.

네이버 또는 지메일 등 전자메일 계정이 있다면 쉽게 깃허브의 계정을 만들 수 있다. 전자메일 계정을 하나 준비해 깃허브의 계정을 만들도록 하자. 깃허브는 내용물을 모두 공개한다면 모두 무료이다.

깃허브 계정 만들기

1	깃허브 홈페이지에 접속해 오른쪽 부분의 Sign up for GitHub 를 누른다.	
2	<p>첫 단계 Create personal account 에서 다음 세 부분을 입력하고 하부 Create an account 를 누른다.</p> <ul style="list-style-type: none"> · Username: 깃허브의 계정으로 사용되는 ID 와 같은 이름 · Email Address: 계정을 만들기 전에 확인 메일을 받기 위한 메일 계정 · Password: 깃허브의 암호 	
3	두 번째 단계 choose your plan 에서는 비용 면에서 계획을 선택하는 단계로 무료로 사용하려면 다음의 처음 버튼을 선택한다.	

- Unlimited private repositories for free: 무료로 공개된 저장소를 무제한 사용 계획

Welcome to GitHub

You've taken your first step into a larger world, @hs7kang.



Completed
Set up a personal account



Step 2:
Choose your plan



Step 3:
Tailor your experience

Choose your personal plan

- ☒ Unlimited public repositories for free.
- ☐ Unlimited private repositories for \$7/month. [\(view in KRW\)](#)

Don't worry, you can cancel or upgrade at any time.

☐ Help me set up an organization next

Organizations are separate from personal accounts and are best suited for businesses who need to manage permissions for many employees.
[Learn more about organizations.](#)

Continue

Both plans include:

- ✓ Collaborative code review
- ✓ Issue tracking
- ✓ Open source community
- ✓ Unlimited public repositories
- ✓ Join any organization

- 4 세 번째 단계 Tailor your experience 에서는 프로그래밍 경험, 사용 계획과 목적, 관심 분야 등을 입력한다. 이 단계는 생략할 수 있으며 작성 후 submit 을 누른다.

Welcome to GitHub

You'll find endless opportunities to learn, code, and create, @hs7kang.

✓ Completed
Set up a personal account

🔧 Step 2:
Choose your plan

⚙️ Step 3:
Tailor your experience

How would you describe your level of programming experience?

☐ Very experienced ☐ Somewhat experienced ☒ Totally new to programming

What do you plan to use GitHub for? (check all that apply)

☐ Research ☐ Development ☐ Design
☐ Project Management ☒ School projects ☐ Other (please specify)

Which is closest to how you would describe yourself?

☒ I'm a hobbyist ☐ I'm a student ☐ I'm a professional
☐ Other (please specify)

What are you interested in?

java python c

e.g. tutorials, android, ruby, web-development, machine-learning, open-source

Submit skip this step

계정 생성이 완료되면 다음 화면이 표시되며, 가입할 때 입력했었던 전자메일 주소로 가입 확인 전자메일이 도착한다. 도착한 메일 내에 버튼을 클릭하여 승인만 해주면 최종 가입이 된다.

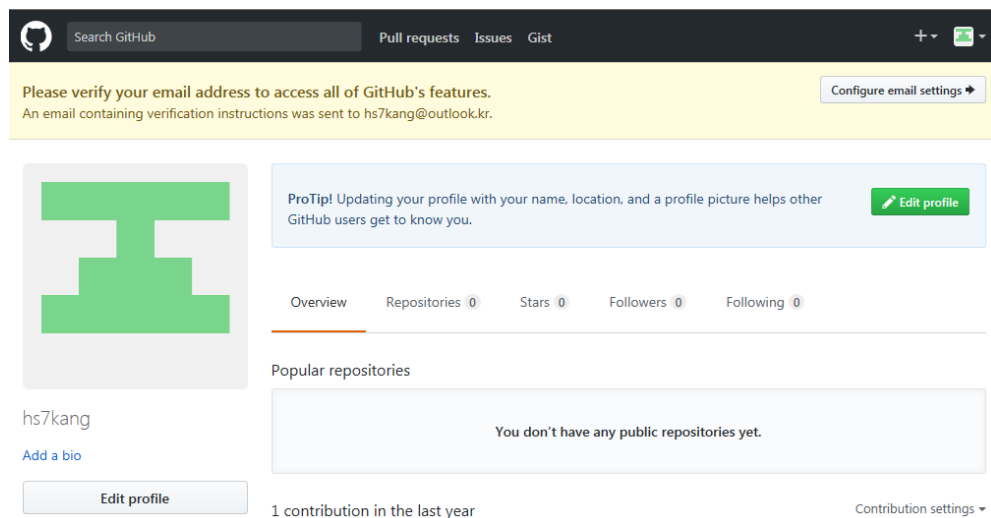
Learn Git and GitHub without any code!

Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request.

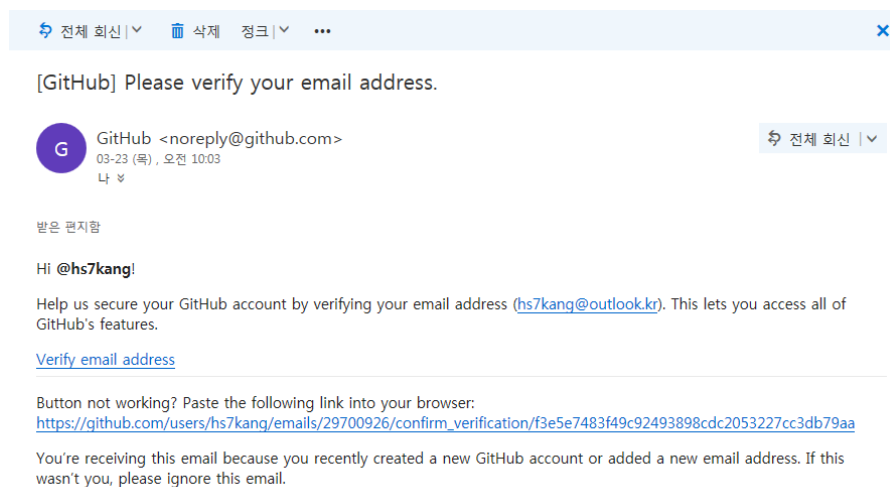
Read the guide Start a project

깃허브 로그인

깃허브 계정 생성에 성공했다라도 전자메일의 확인이 없이 깃허브에 로그인(sign in)하면 다음과 같이 메시지가 표시되며 사용이 일부 제한적이다.

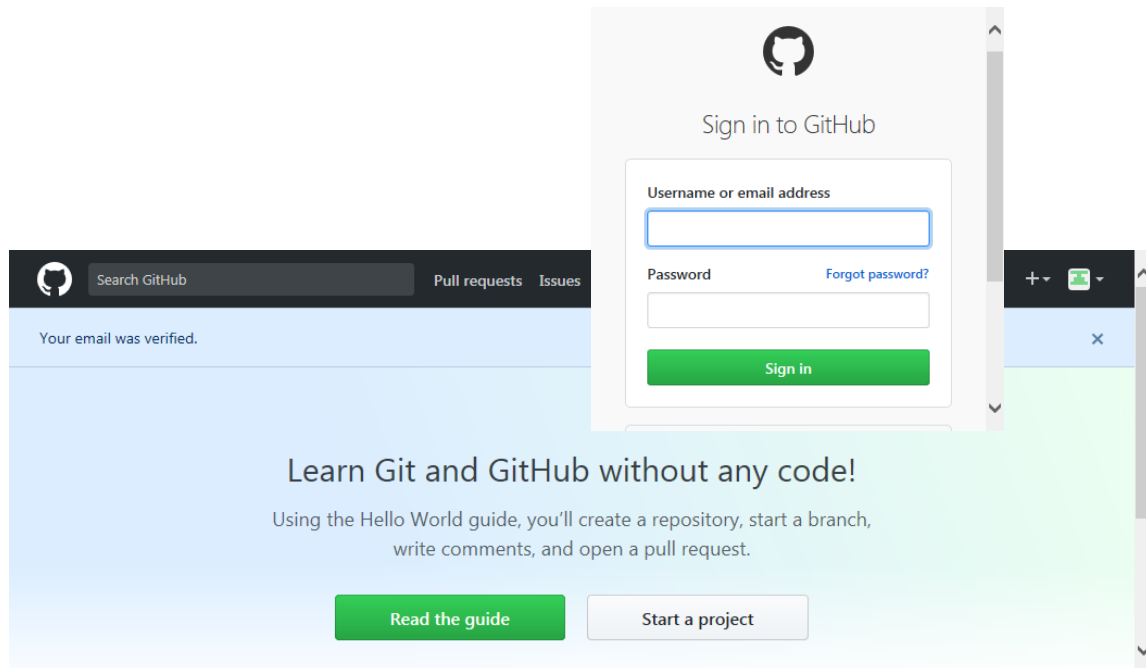


깃허브 계정 생성 확인을 위해 깃허브에 입력한 메일을 확인하면 다음과 같이 확인 메일을 볼 수 있다.



위 메일 내용의 **Verify email address** 를 누르면 깃허브 로그인 페이지로 이동하며, 사용자이름과 암호로 로그인을 수행한다.

- **Username or email address:** 깃허브의 계정이나 전자주소를 입력



로그인 성공 후 이동된 페이지의 왼쪽 상단에서 [Your email was verified]을 확인할 수 있으며, 이제 두 개의 버튼으로 깃허브의 작업을 수행할 수 있다. 버튼 **Read the guide** 로 간단히 깃허브의 주요 작업을 안내하는 “Hello World” 수행 방법을 알 수 있으며, **Start a Project** 로 필요한 저장소의 생성을 직접 수행할 수 있다.

Tips:

깃허브의 가이드에서 깃허브 정의를 살펴보면 다음과 같다.

“깃허브는 버전 제어와 공동 작업을 위한 코드 관리 플랫폼이다. 우리는 어디에서나 다른 사람과 함께 프로젝트를 수행할 수 있다.”

GitHub is a code hosting platform for version control and collaboration. It lets you and others work together on projects from anywhere.

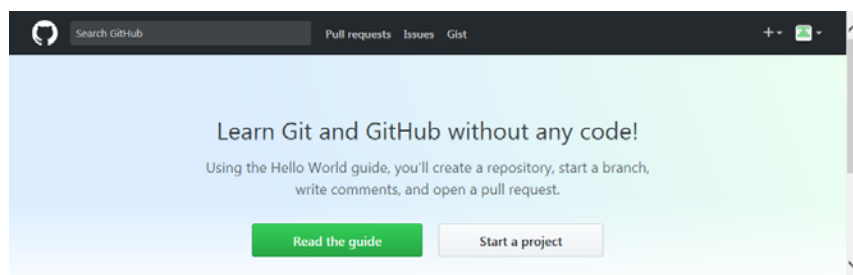
3 파일 저장과 버전 제어

간단한 소스를 저장하고 버전 제어를 직접 경험 하자

지스트 이용

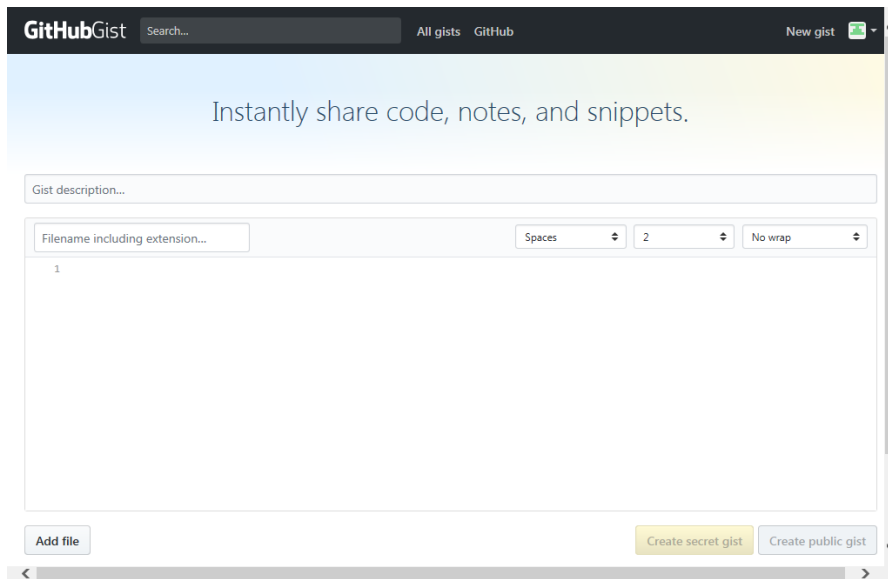
로그인한 깃허브 화면에서 지스트(요지, 핵심)를 사용하여 간단한 코드나 노트를 저장하고 공유할 수 있다.

- **Gists:** 작업을 공유하는 방법으로 단일 파일, 파일의 일부 또는 전체 응용 프로그램을 공유 가능
- 지스트만을 접속하려면 <https://gist.github.com>



다음 지스트 편집화면에서 적절한 제목과 파일이름을 입력한다.

- **Gist description...:** 여러 개의 소스 코드의 모임 이름
- **Filename including extensions...:** 확장자를 포함한 소스 파일 이름



지스트의 사용방법과 파일의 버전관리를 알아 보자.

1. 설명과 파일이름을 입력한 후 소스를 코딩한다. 프로그래밍 소스라면 다음과 같이 확장자 **java** 를 붙여야 소스의 색상이 표시된다. 입력 후 **Create public gist** 로 지스트를 생성한다.



2. 다음과 같이 소스가 생성되면 설명을 부가적으로 쓰고 **Comment** 를 누른다.

자바 기초 프로그래밍

BitOp.java

Raw

```

1  import java.util.Scanner;
2
3  public class BitOp {
4      public static void main(String[] args) {
5          // TODO Auto-generated method stub
6          Scanner input = new Scanner(System.in);
7          //이진수 입력
8          int binary = input.nextInt(2);
9          input.close();
10
11         System.out.println(binary);
12     }
13 }

```

Write

Preview

AA B i “ < > ☰ ☷ ☹ ↶ @

이진수 입력 후 십진수로 출력

Attach files by dragging & dropping or [selecting them](#).

Styling with Markdown is supported

Comment

3. 다음과 같이 소스와 설명이 표시된다.

자바 기초 프로그래밍

BitOp.java

Raw

```

1  import java.util.Scanner;
2
3  public class BitOp {
4      public static void main(String[] args) {
5          // TODO Auto-generated method stub
6          Scanner input = new Scanner(System.in);
7          //이진수 입력
8          int binary = input.nextInt(2);
9          input.close();
10
11         System.out.println(binary);
12     }
13 }

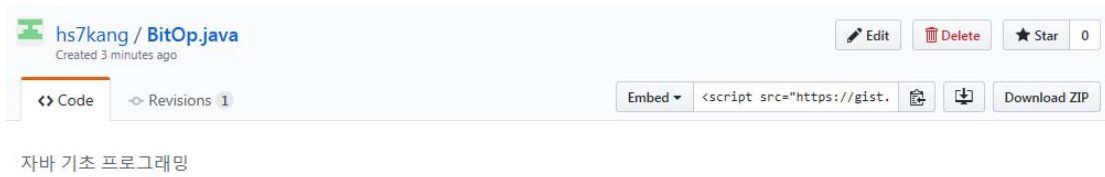
```

hs7kang commented 39 seconds ago

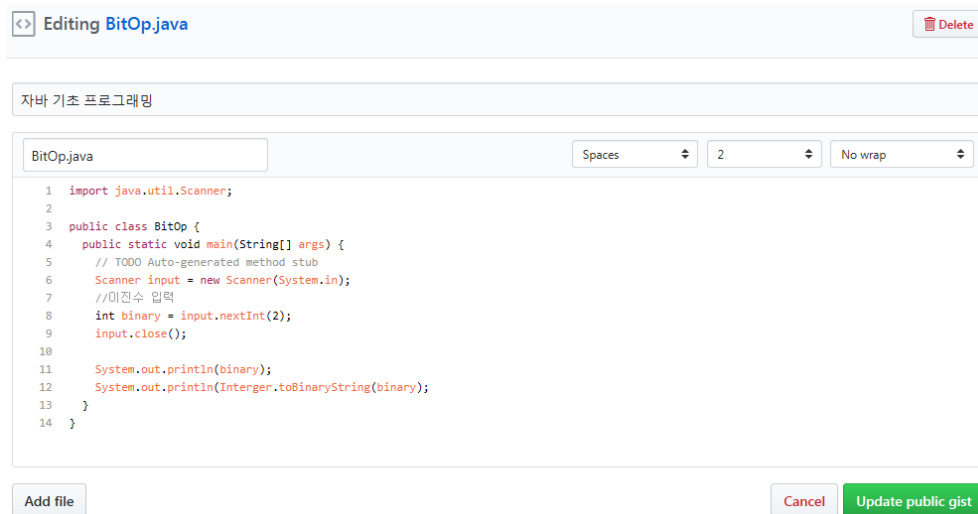
Owner ✎ ✕

이진수 입력 후 십진수로 출력

4. 편집한 지스트를 다시 수정하려면 파일의 메뉴 Edit 를 누른다.

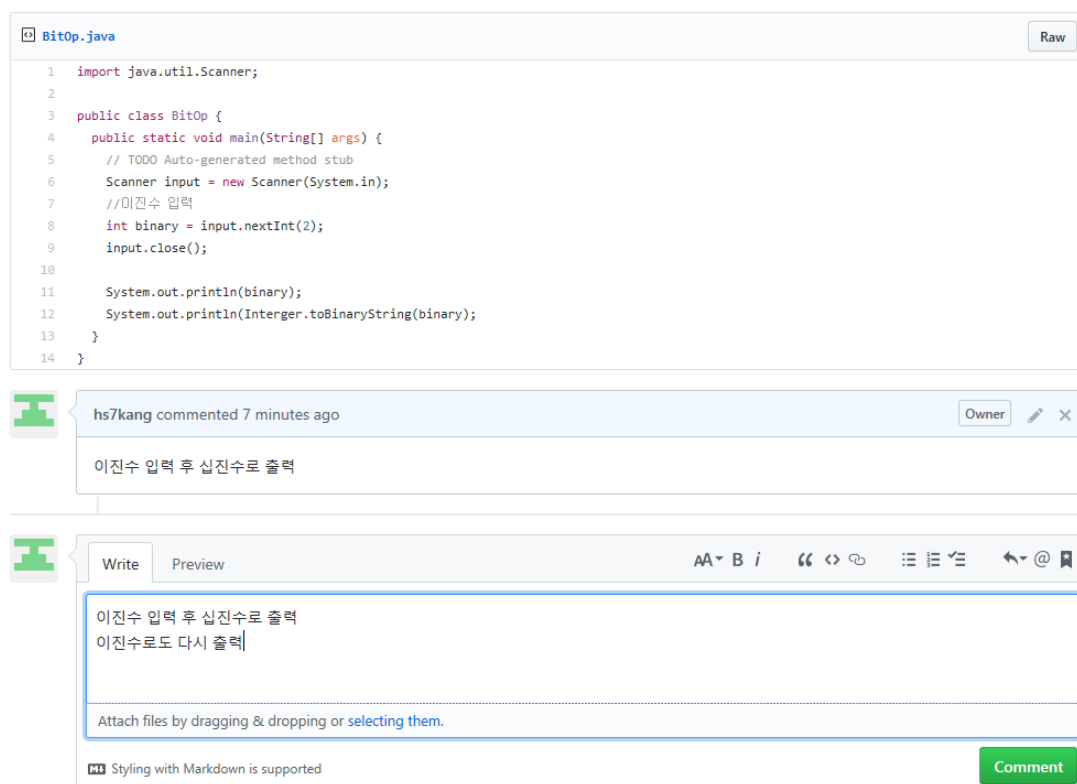


5. 지스트에서 필요한 부분을 다시 편집한 후 Update public gist 를 누른다.



6. 다시 수정된 파일의 설명을 입력한 후 Comment 를 누른다.

자바 기초 프로그래밍



The screenshot shows a Gist interface. At the top, there's a tab for 'BitOp.java' with a 'Raw' button. Below it is a code editor containing the following Java code:

```


1 import java.util.Scanner;
2
3 public class BitOp {
4     public static void main(String[] args) {
5         // TODO Auto-generated method stub
6         Scanner input = new Scanner(System.in);
7         //이진수 입력
8         int binary = input.nextInt(2);
9         input.close();
10
11         System.out.println(binary);
12         System.out.println(Integer.toBinaryString(binary));
13     }
14 }

```

Below the code, there's a comment box. It shows a comment from 'hs7kang' made 7 minutes ago. The comment text is: '이진수 입력 후 십진수로 출력'.

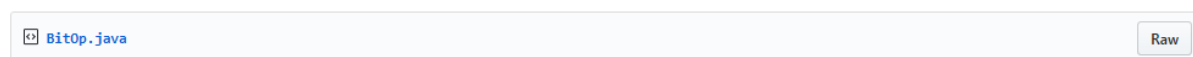
Below the comment, there's a 'Write' tab and a 'Preview' tab. The 'Write' tab is active, showing a text area with the same comment text: '이진수 입력 후 십진수로 출력' and '이진수로도 다시 출력'. Below the text area is a placeholder text: 'Attach files by dragging & dropping or selecting them.' and a 'Comment' button.

7. 처음 생성 후 1 번 수정한 자바 소스 파일 BitOp.java 의 Revisions 를 보면 파일의 생성과 수정한 수인 2 로 표시되는데 파일의 구체적인 변화를 보려면 눌러 확인할 수 있다.



The screenshot shows the Gist interface for 'hs7kang / BitOp.java'. At the top, there's a header with the user name 'hs7kang', the file name 'BitOp.java', and the status 'Last active 2 minutes ago'. To the right of the header are buttons for 'Edit', 'Delete', 'Star', and '0'. Below the header, there's a tab for 'Code' and a tab for 'Revisions 2'. The 'Revisions' tab is active, showing a list of revisions. Below the revisions list, there's a section for 'Embed' with a dropdown menu and a script tag: '<script src="https://gist.' To the right of the script tag are buttons for 'Embed', 'Download ZIP', and 'Download ZIP'.

자바 기초 프로그래밍



The screenshot shows a Gist interface with a tab for 'BitOp.java' and a 'Raw' button.

8. 다음은 파일 수정 내역이 보이는 화면으로, 가장 최근에 수정된 내용이 먼저 표시된다. 녹색 바탕과 줄 번호 다음의 + 표시는 추가된 줄임을 알 수 있다.

- 3분전에 수정된 파일에서는 한 줄이 추가된 것을 표시
- 11분전에 만들어진 첫 파일에서는 전체 13 줄이 모두 추가된 것을 표시

The screenshot displays the 'Revisions' section of a GitHub Gist for the file `BitOp.java`. The top revision, made 3 minutes ago, shows a diff where line 12 was added. A red dashed box highlights the diff header and the new line. A callout box explains that the '1' indicates the number of added lines, and the green color indicates the relative amount of added lines. The bottom revision, created 11 minutes ago, shows the full file content with lines 1 through 13 highlighted in green, indicating they were all added.

1은 추가된 줄의 수이며, 녹색의
높음은 추가된 줄의 상대적인 양을
표시한다.

9. 파일에서 11 줄을 지우고 다시 수정하기 위해 Update public gist 를 누른다.

자바 기초 프로그래밍

BitOp.java Spaces 2 No wrap

```

1 import java.util.Scanner;
2
3 public class BitOp {
4     public static void main(String[] args) {
5         // TODO Auto-generated method stub
6         Scanner input = new Scanner(System.in);
7         //이진수 입력
8         int binary = input.nextInt(2);
9         input.close();
10
11         System.out.println(Integer.toBinaryString(binary));
12     }
13 }

```

Add file Cancel Update public gist

10. 한 줄 삭제된 수정에 대한 주석을 다시 입력하고 Comment 를 누른다.

자바 기초 프로그래밍

BitOp.java Raw

```

1 import java.util.Scanner;
2
3 public class BitOp {
4     public static void main(String[] args) {
5         // TODO Auto-generated method stub
6         Scanner input = new Scanner(System.in);
7         //이진수 입력
8         int binary = input.nextInt(2);
9         input.close();
10
11         System.out.println(Integer.toBinaryString(binary));
12     }
13 }

```

hs7kang commented 13 minutes ago Owner ✎ ✕

이진수 입력 후 십진수로 출력

hs7kang commented 4 minutes ago Owner ✎ ✕

이진수 입력 후 십진수로 출력
이진수로도 다시 출력

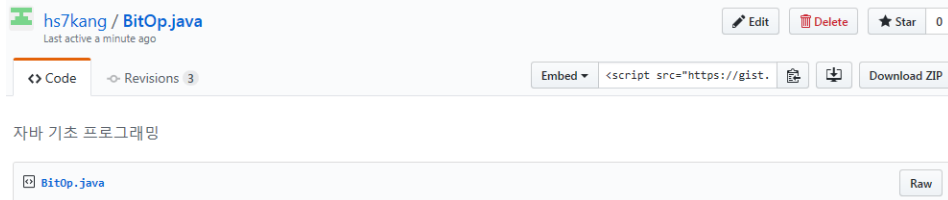
Write Preview AA B i “ < > ☰ ☷ ☹ ↶ @ 📌

이진수 입력 후 이진수로 바로 출력

Attach files by dragging & dropping or [selecting them](#).

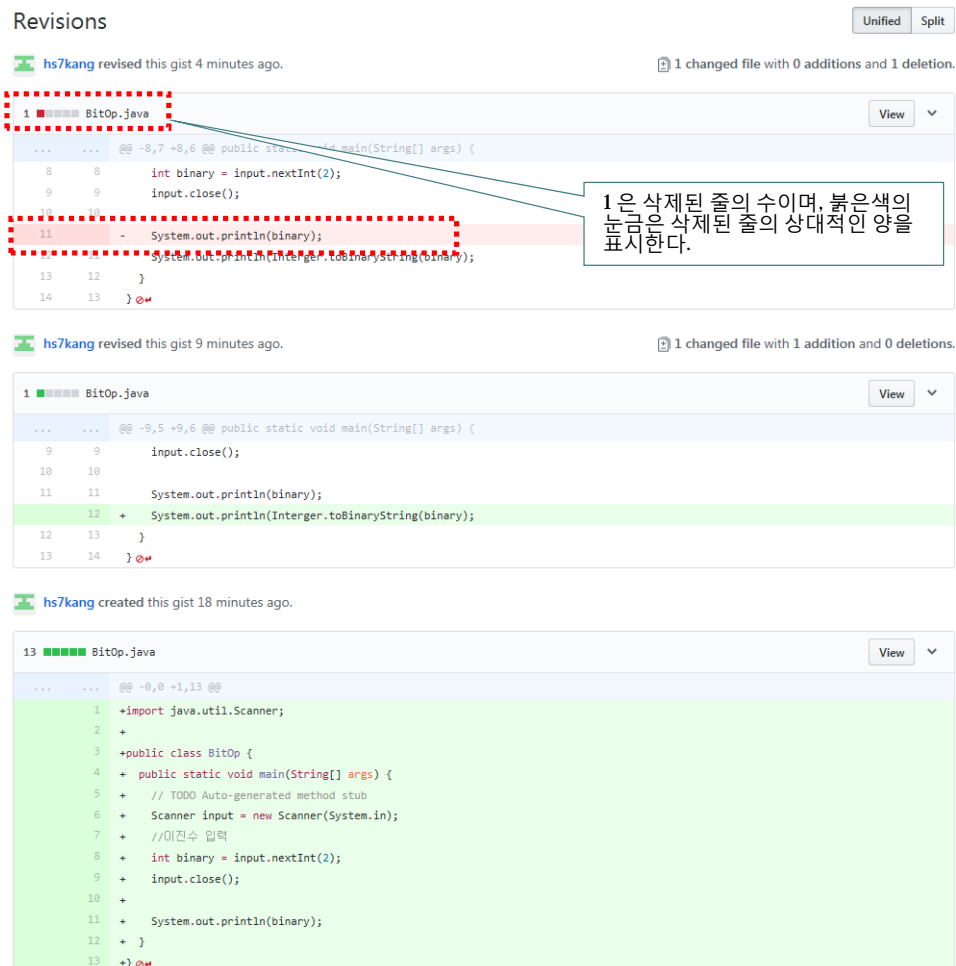
Styling with Markdown is supported Comment

11. 이제 파일 BitOp.java 의 Revisions 수인 3 을 누른다.



12. 파일 BitOp.java 의 3 회인 수정 이력이 간결히 표시된다. 붉은 바탕과 줄 번호 다음의 - 표시는 삭제된 줄임을 알 수 있다.

- 4 분전에 수정된 파일에서는 한 줄이 삭제된 것을 표시



13. 파일의 수정내역이 표시된 부분에 마우스를 가져가면 다음과 같이 추가와 삭제된 줄의 개수가 표시된다.

 **hs7kang** revised this gist 5 minutes ago.

```
1 0 additions & 1 deletion
... @@ -8,7 +8,6 @@ public static void main(String[] args) {
8      8      int binary = input.nextInt(2);
9      9      input.close();
10     10
11     - System.out.println(binary);
12     11      System.out.println(Integer.toBinaryString(binary);
13     12      }
14     13      } 〇
```

4 저장소와 파일 생성

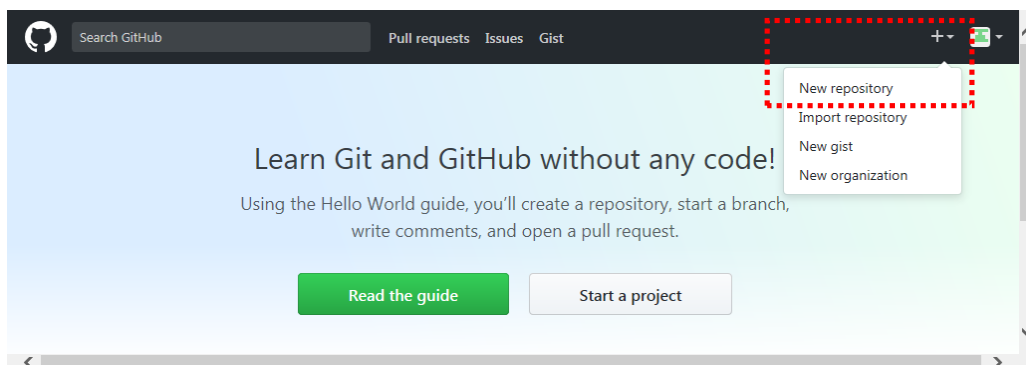
깃허브 저장소 생성과 파일 생성

깃허브에 저장소 만들기

이제 깃허브에서 저장소(**Repository**)를 만들어 보자. 저장소는 프로젝트 관련 파일이 모여 있는 폴더와 같다.

- 저장소는 수행하는 프로젝트 관련 파일이 저장되는 장소이며, 각 파일의 수정 이력도 저장

깃허브에서 저장소를 만들려면 깃허브에 로그인하여 우측 상단의 상단의 +를 눌러 메뉴 [**New repository**]를 선택하여 시작한다.



저장소의 이름을 **Repository name** 에 입력한 후 **Initialize this repository with a README** 를 체크한 후 [**Create repository**]를 눌러 깃허브의 저장소를 생성한다. 저장소의 설명을 **Description** 에 입력할 수 있으며, **Public** 을 선택하면 공개적으로 원격저장소를 무료로 웹 서비스할 수 있으며, 비공개로 목적으로 **Private** 으로 선택하면 비용을 지불해야 한다.

Search GitHub Pull requests Issues Gist

Create a new repository

A repository contains all the files for your project, including the revision history.

Owner: **hs7kang** / Repository name: **Java Programming** ✓

Great repository name: Your new repository will be created as **Java-Programming** redesigned-train.

Description (optional): **자바 프로그래밍 연습**

☒ **Public**
Anyone can see this repository. You choose who can commit.

☐ **Private**
You choose who can see and commit to this repository.

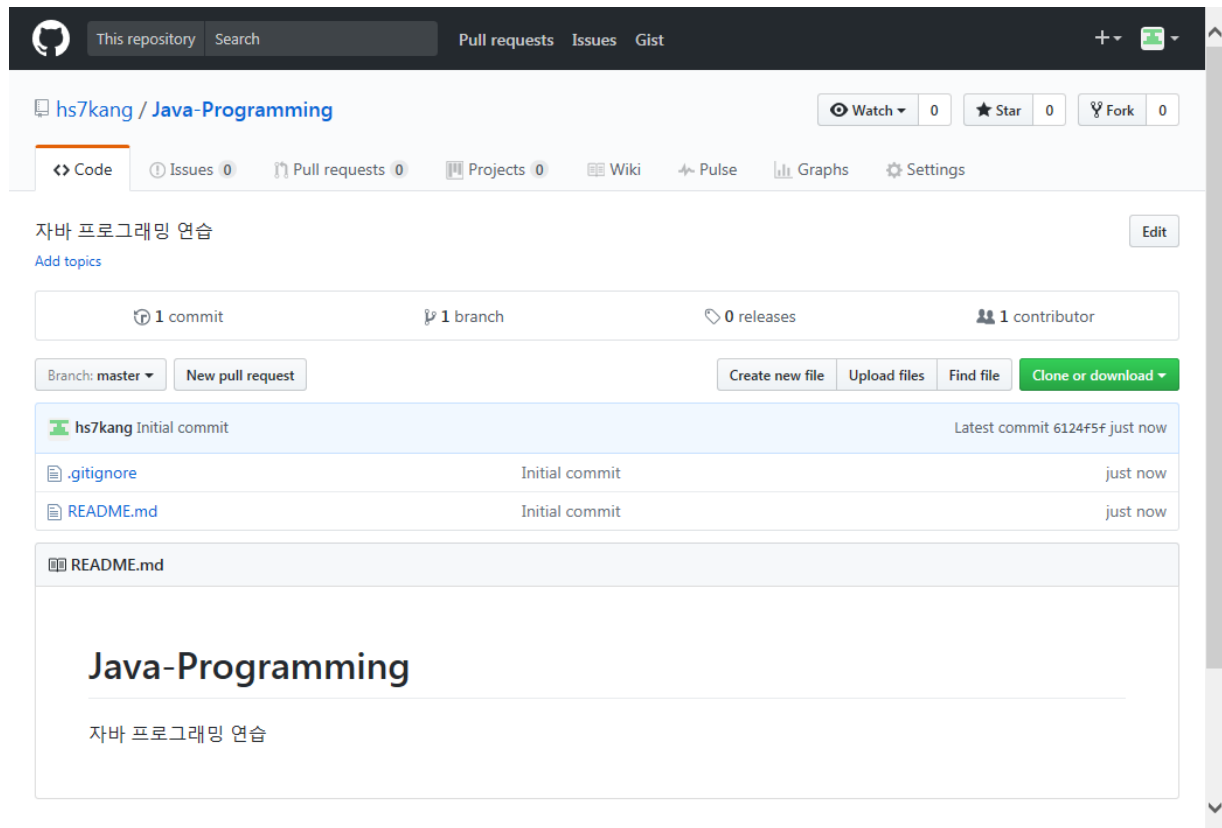
☒ **Initialize this repository with a README**
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: **Java** Add a license: **None** ⓘ

Create repository

다음 화면은 저장소 'Java Programming'이 생성된 화면이다. 다음은 사용자 계정 **hs7kang**으로 깃허브 저장소 이름 'Java Programming'이 생성된 저장소이다. 이제 생성된 저장소는 브라우저에서 주소 <https://github.com/hs7kang/Java-Programming>으로 접속하면 누구나 저장소에 직접 접근할 수 있다. 깃허브 저장소 이름에서 빈 공간은 주소에서 '-'로 대체되는 것을 알 수 있다. 친구들의 깃허브에 접속해 보도록 하자.

- <https://github.com/hs7kang/Java-Programming>
- <https://github.com/사용자이름/저장소이름>

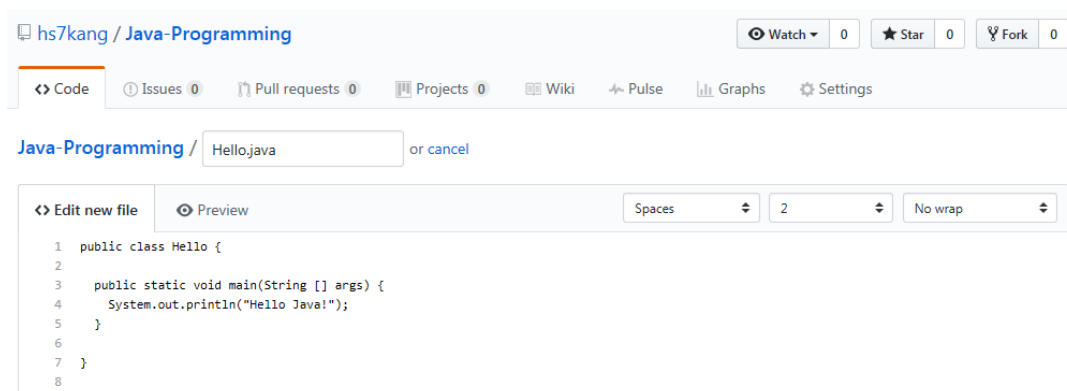


저장소에 파일 만들기

저장소에 파일을 저장하는 방법은 파일을 직접 만들거나 아니면 데스크 탑의 파일을 올리는 방법이다.

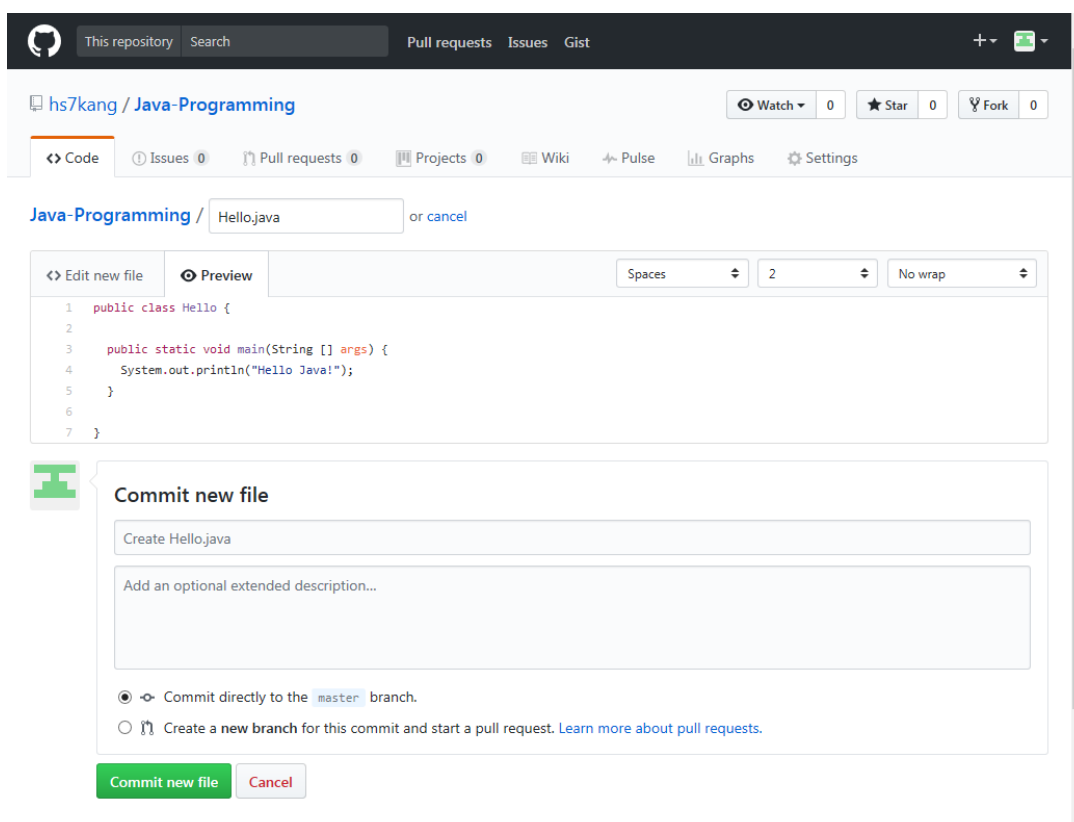
- **Create new file:** 편집기를 이용해 직접 파일을 작성
- **Upload files:** 자신의 데스크탑에서 파일을 업로드

메뉴 **Create new file** 을 누르면 파일을 편집할 수 있는 화면이 표시된다. 파일이름을 쓰고 편집을 완료한다. 파일이 프로그램 소스라면 확장자를 정확히 입력하여 소스 모습을 보기 좋게 할 수 있다.

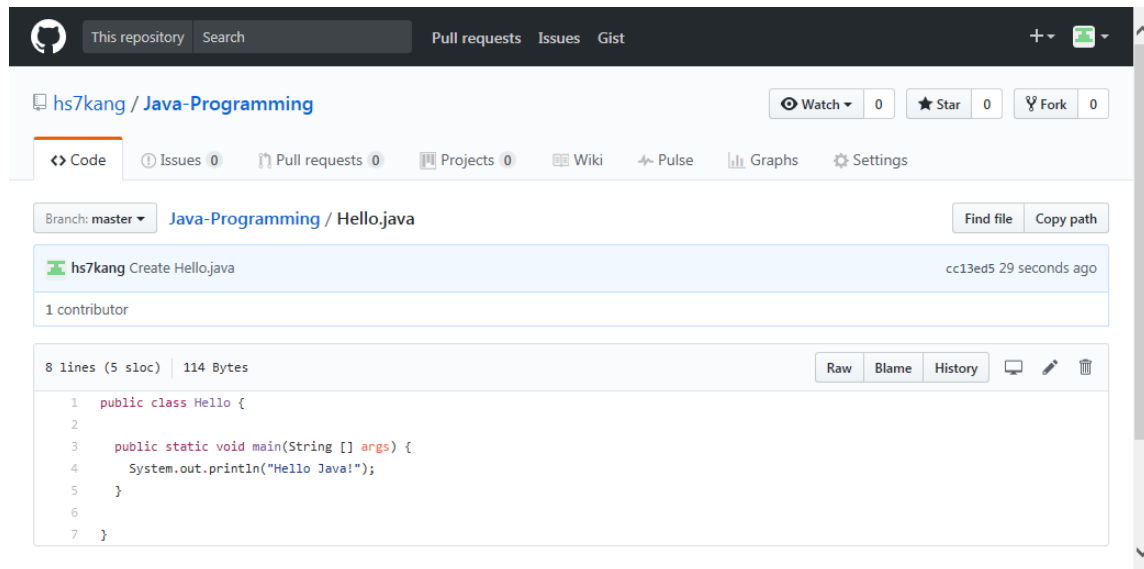


탭 **Preview** 를 누르면 소스의 색상이 표시되며, 필요한 설명을 입력한 후 **Commit new file** 로 파일을 생성한다.

- **Commit:** 커밋은 저장소의 파일을 이력을 관리하기 위해 저장하는 단위



생성된 파일 **Hello.java** 를 누르면 다음과 같이 파일 내용이 표시된다.




저장된 파일에 대한 상세 보기 메뉴는 **Raw, Blame, History** 세 가지이다.

- **Raw:** 브라우저에서 단순한 포맷으로 파일 보이기

```
public class Hello {  
  
    public static void main(String [] args) {  
        System.out.println("Hello Java!");  
        System.out.println("Hello World!");  
    }  
}
```

- **Blame:** 파일의 행에 대한 변경 사항을 추적하고 시간 경과에 따라 파일의 일부분이 어떻게 전개되는지 확인 가능

Java-Programming / Hello.java 

100644 | 9 lines (6 sloc) | 154 Bytes

Raw Normal view History

18 hours ago

```

1 public class Hello {
2
3 public static void main(String [] args) {
4     System.out.println("Hello Java!");
5     System.out.println("Hello World!");
6 }
7
8 }

```

줄 별로 생성되거나 수정된 커밋 시점을 보이고 있다.

- **History:** 파일에 대한 수정 커밋에 대한 이력을 표시

History for Java-Programming / Hello.java

Commits on Apr 3, 2017

Update Hello.java ...
hs7kang committed on GitHub 11 minutes ago

4ba6271

Commits on Apr 2, 2017

Create Hello.java
hs7kang committed on GitHub 19 hours ago

cc13ed5

최근 커밋 이력부터 표시된다.

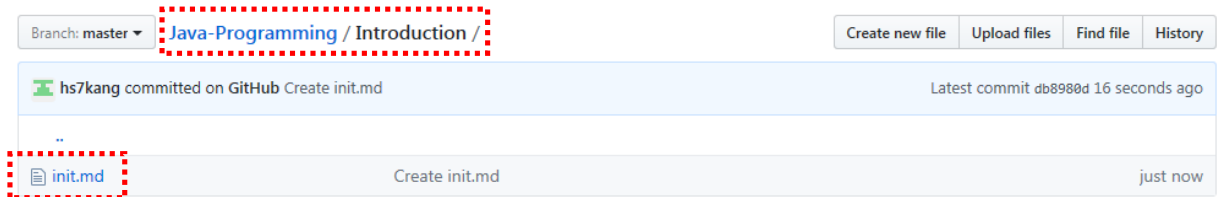
저장소에 직접 폴더 만들기

저장소에 직접 폴더를 만들고 싶다면 파일이름 입력 부분에 원하는 폴더의 이름을 입력하고 그 뒤에 /를 입력한 후 파일이름을 입력한다. / 이 입력되는 순간 바로 폴더의 모습으로 바뀌는 것을 알 수 있다. 폴더만을 만들 수는 없으니 파일을 하나 만들도록 한다. 즉 다음과 같이 입력하면 하부 그림처럼 표시된다.

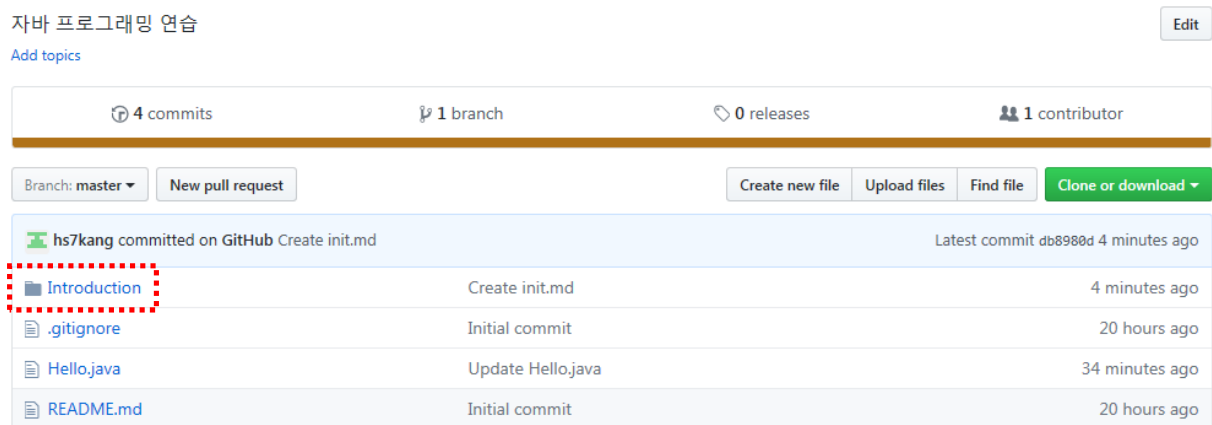
- Introduction/init.md

Java-Programming / Introduction / or cancel

하단 Commit new file 을 누르면 다음과 같이 폴더 Introduction 과 파일 init.md 가 생성된 것을 확인할 수 있다.



다음은 저장소에서 하부 폴더 **Introduction** 과 그 하부 파일을 본 화면이다. 폴더 **Introduction** 을 누르면 하부로 이동할 수 있다.



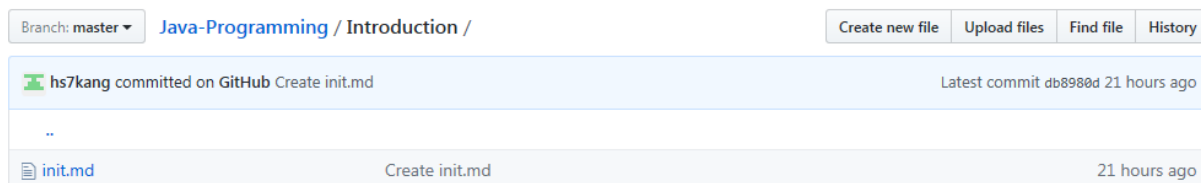
Tip:

파일입력 부분에서 직접 상위 디렉토리로 이동하려면 파일 이름 필드의 시작 부분에 커서를 놓고 **../** 또는 **backspace** 를 누른다.

· **../** 또는 **backspace**

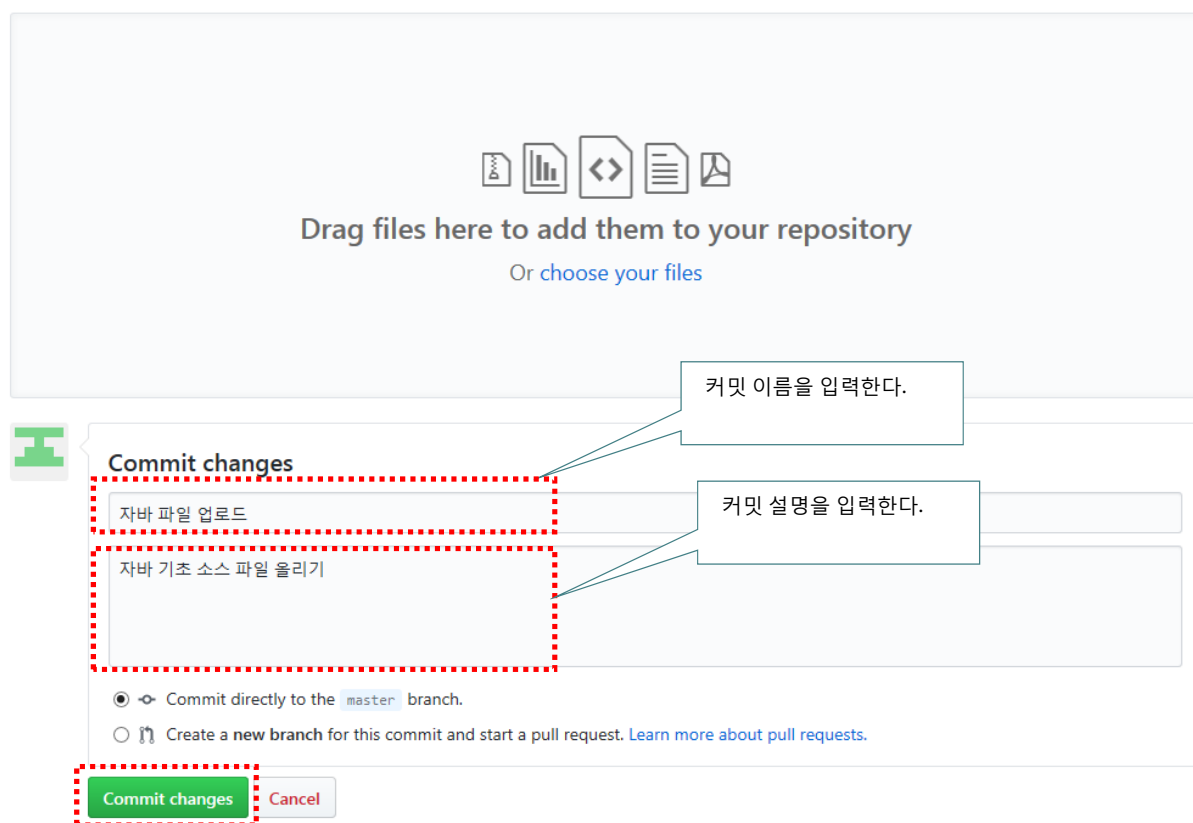
저장소에 파일 업로드

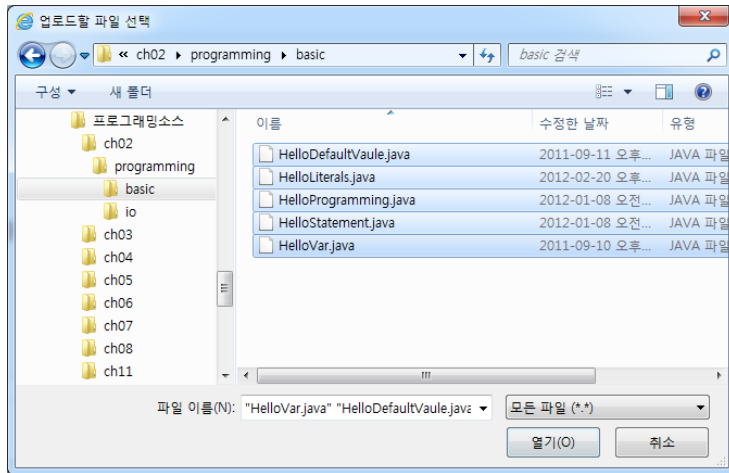
데스크탑이나 이동 저장장치의 파일 업로드를 위해 지정 저장소의 위치에서 메뉴 **Upload files** 를 누른다.



표시된 다음 화면에서 'Choose your files'을 눌러 업로드할 파일을 선택하거나 파일을 드래그 드롭한다.
파일 업로드의 커밋을 저장하기 위해 가능하면 커밋이름과 설명을 입력한다.

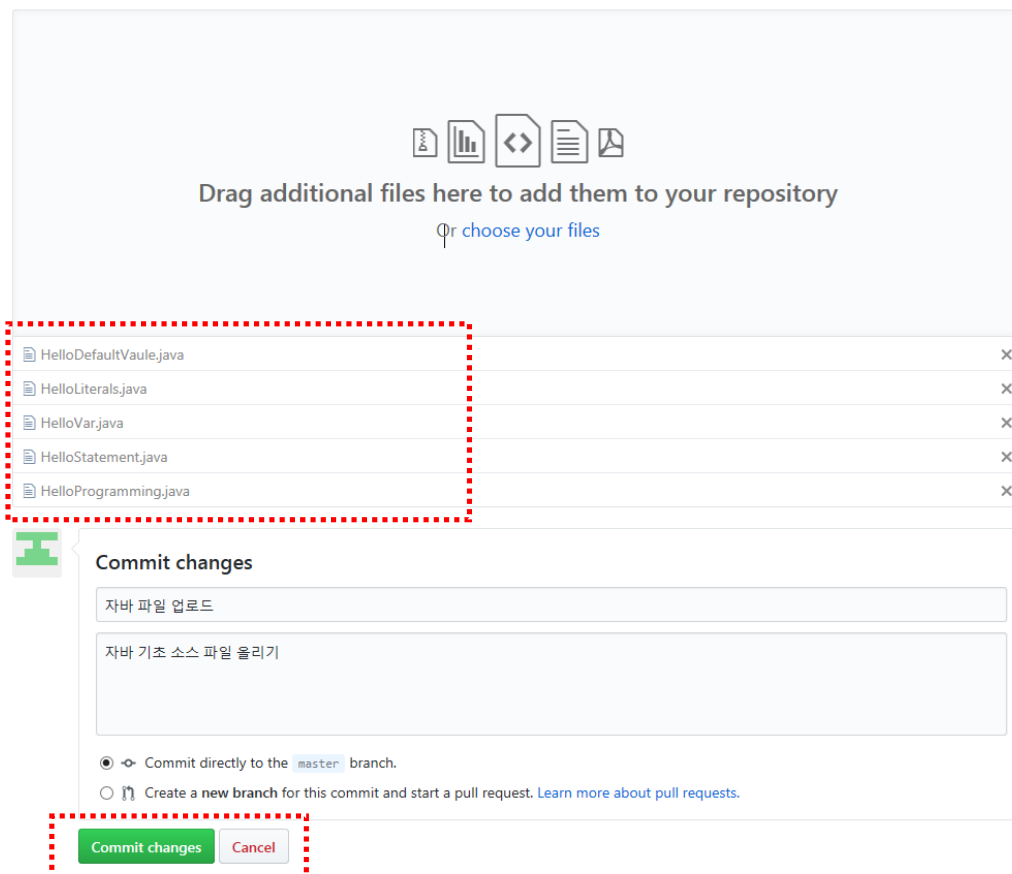
Java-Programming / Introduction





다음은 업로드할 파일이 선택된 모습이다. 이제 파일이 선택된 것을 확인하고 하부의 'Commit changes' 버튼을 누른다.

Java-Programming / Introduction



이제 다음 화면에서 저장소 'Open-Source'에 업로드된 파일을 볼 수 있다.

Branch: master ▾	Java-Programming / Introduction /	Create new file	Upload files	Find file	History
<div> hs7kang committed on GitHub 자바 파일 업로드 ... Latest commit 8ba2acb 14 seconds ago </div>					
..					
HelloDefaultVaule.java	자바 파일 업로드	14 seconds ago			
HelloLiterals.java	자바 파일 업로드	14 seconds ago			
HelloProgramming.java	자바 파일 업로드	14 seconds ago			
HelloStatement.java	자바 파일 업로드	14 seconds ago			
HelloVar.java	자바 파일 업로드	14 seconds ago			
init.md	Create init.md	21 hours ago			

위 파일 목록에서 파일 하나를 선택하여 누르면 소스 파일을 다음과 같이 볼 수 있다. 파일의 우측 상단에는 Raw, Blame, History 등 사용 메뉴를 볼 수 있으며, 만일 이 파일을 삭제하고 싶다면 가장 오른쪽 쓰레기 버튼을 사용한다.

Branch: master ▾
Java-Programming / Introduction / HelloDefaultVaule.java
Find file
Copy path

hs7kang 자바 파일 업로드
8ba2acb 2 minutes ago

1 contributor

18 lines (14 sloc) | 316 Bytes

Raw
Blame
History

```

1 package programming.basic;
2
3 public class HelloDefaultVaule {
4     //필드 선언
5     static int def;
6     static boolean bool;
7
8     public static void main(String[] args) {
9         //지역변수 선언
10        int n = 100;
11        System.out.println(n);
12
13        //필드 사용 가능
14        System.out.println(def);
15        System.out.println(bool);
16    }
17 }

```

5 깃허브 데스크탑을 활용한 깃허브 저장소 관리

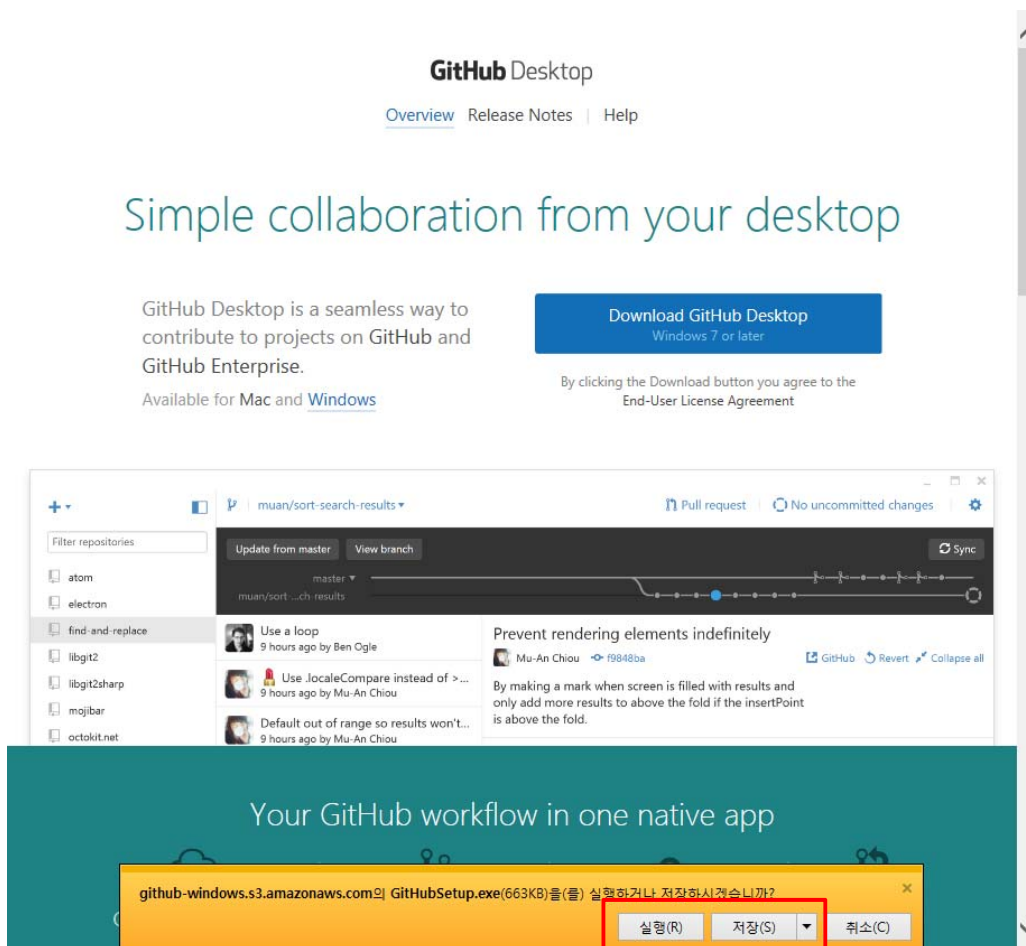
깃허브 데스크탑 설치와 실행

깃허브 데스크탑 설치

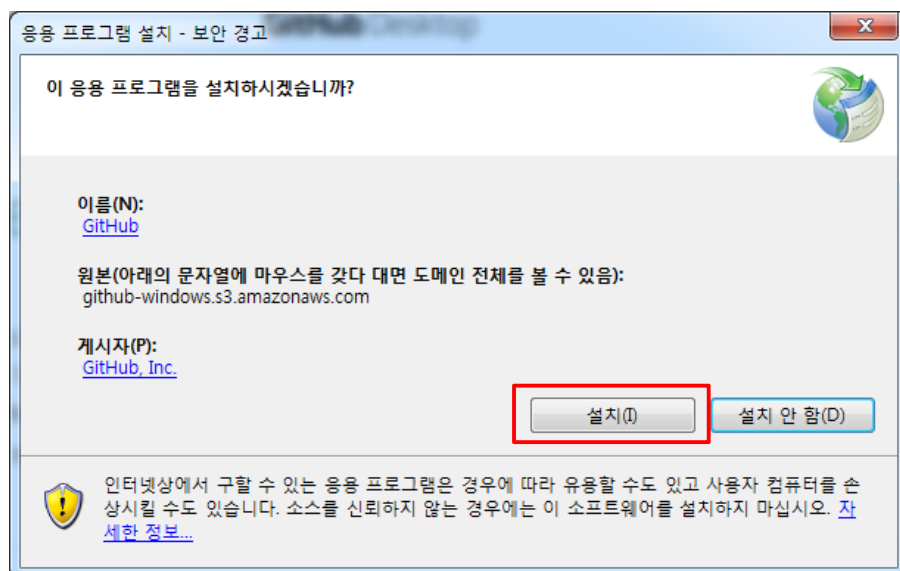
다음 깃허브 데스크탑 사이트에 접속하여 깃허브 데스크탑 설치 파일을 내려 받자.

- desktop.github.com

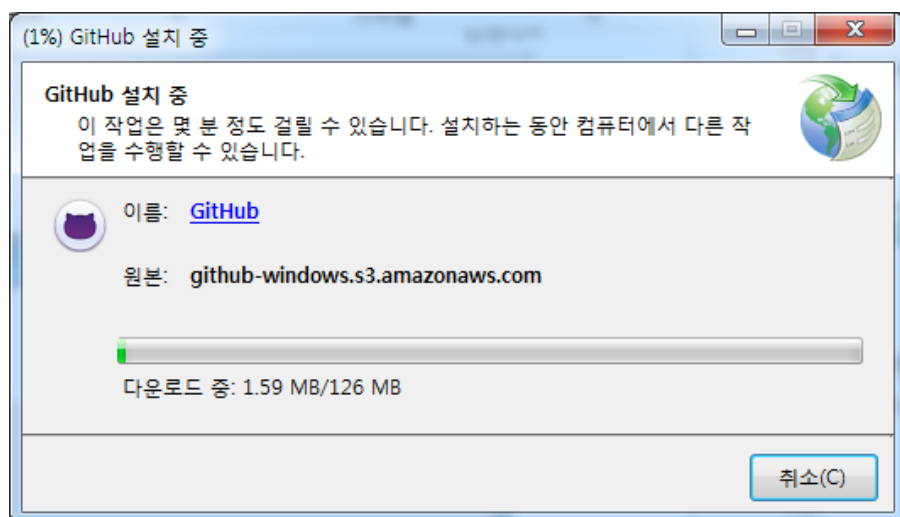
접속한 화면에서 [Download GitHub Desktop]을 눌러 깃허브 데스크탑을 설치한다.



위 하단 [실행] 버튼을 눌러 설치를 시작하면 다음 화면에서 [설치]를 누른다.

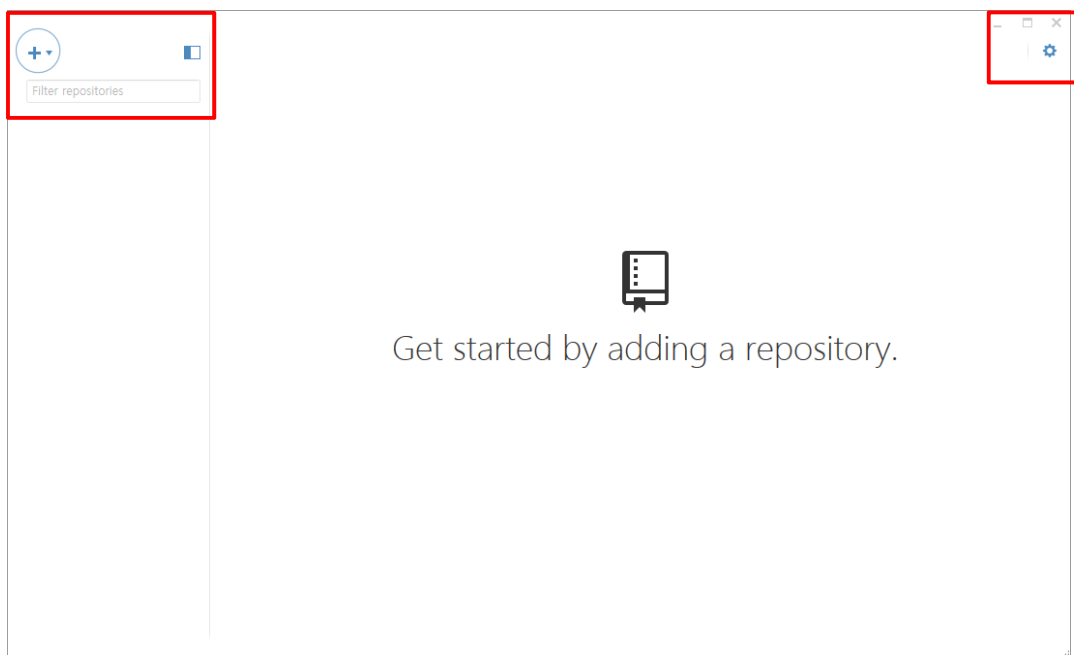


이제 다음 설치 중 화면이 표시되며 필요한 파일을 내려 받아 설치한다.



깃허브 데스크탑 실행

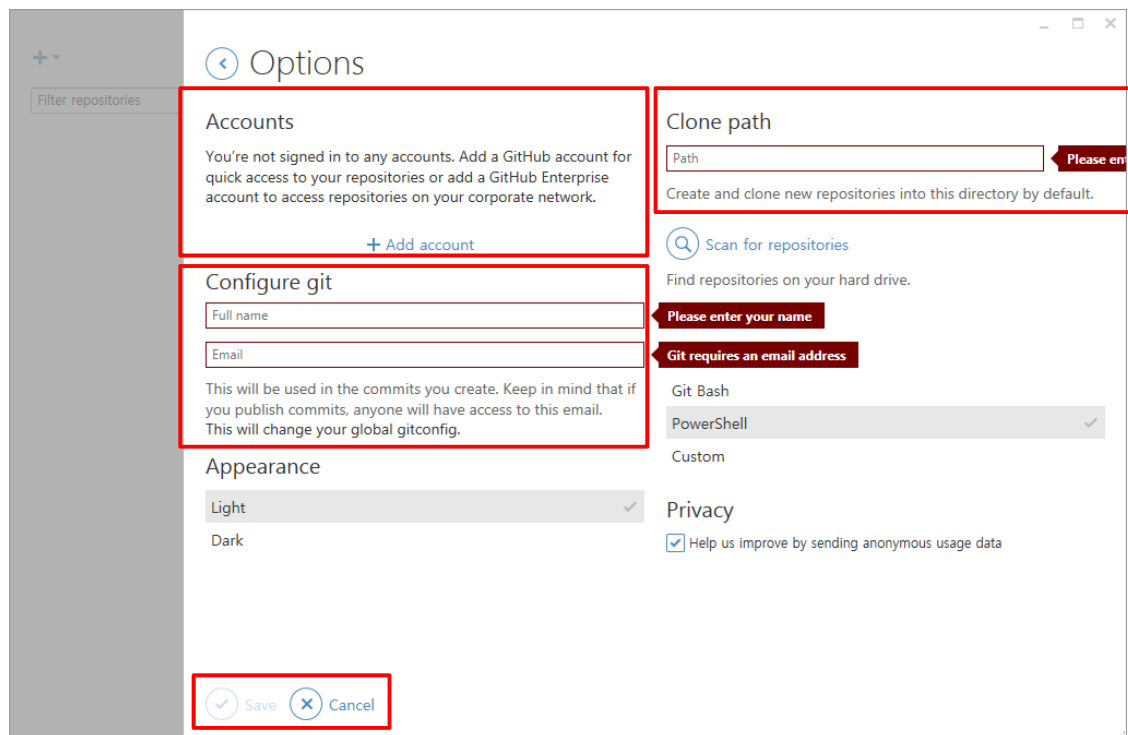
깃허브 데스크탑을 실행하면 다음과 같이 매우 간단한 화면이 실행된다. 화면은 크게 좌측과 우측으로 비어 있는 두 부분이 나뉘어 구성되며 왼쪽 열을 보이지 않게 하려면 상단 중간 버튼을 눌러 없앨 수 있다. 좌측 메뉴 +은 저장소 관리 확장 메뉴이며, 오른쪽 메뉴는 깃허브 데스크탑의 옵션 메뉴로 여러 환경 설정을 위한 메뉴이다.



깃허브 데스크탑 환경 설정

화면 우측 상단의 [옵션] 버튼을 누르면 다음 화면이 표시된다. 옵션 화면은 **Accounts**, **Configure git**, **Clone Path** 등의 정보를 지정한다.

- **Accounts:** 자신의 깃허브 계정을 추가
- **Configure git:** 자신의 사용자와 전자우편 주소를 입력
- **Clone path:** 자신의 컴퓨터의 지역저장소로 사용하려는 상위 폴더를 하나 지정



깃허브에서 관련 파일을 저장하는 저장소는 자신의 컴퓨터 저장소인 지역저장소와 깃허브의 원격저장소로 나눌 수 있다.

- 지역저장소(Local Repository): 자신의 컴퓨터에서 작업을 수행하기 위해 지정한 저장소
- 원격저장소(Remote Repository): 지역저장소와 대응하는 깃허브의 저장소

다음은 깃허브 계정을 추가하는 화면으로 깃허브에 등록된 이메일 주소와 암호를 입력한 후 Log in 버튼을 누른다.

◀ Log in

GitHub GitHub Enterprise


The best way to build and ship software. [Go to github.com](https://github.com) to sign up for an account

☒ Log in ☐ Cancel

로그인이 성공하면 다음과 같이 **Accounts** 에 자신 ID 정보를 볼 수 있으며, 특히 자신의 컴퓨터의 지역저장소로 사용하려는 상위 폴더인 **Clone path** 를 하나 지정하도록 한다. 다음을 참고로 적절히 다른 옵션을 모두 지정한 후 하부 **Save** 버튼을 누른다.

Options

Accounts

 **hs7kang**
hs7kang

Free plan (no private repositories) [Manage](#)

[+ Add GitHub Enterprise account](#)

Clone path

[Browse](#)

Create and clone new repositories into this directory by default.

Configure git

This will be used in the commits you create. Keep in mind that if you publish commits, anyone will have access to this email. This will change your global gitconfig.

Appearance

☒ Light ☐ Dark

Default shell

Cmd

Git Bash

PowerShell ✓

Custom

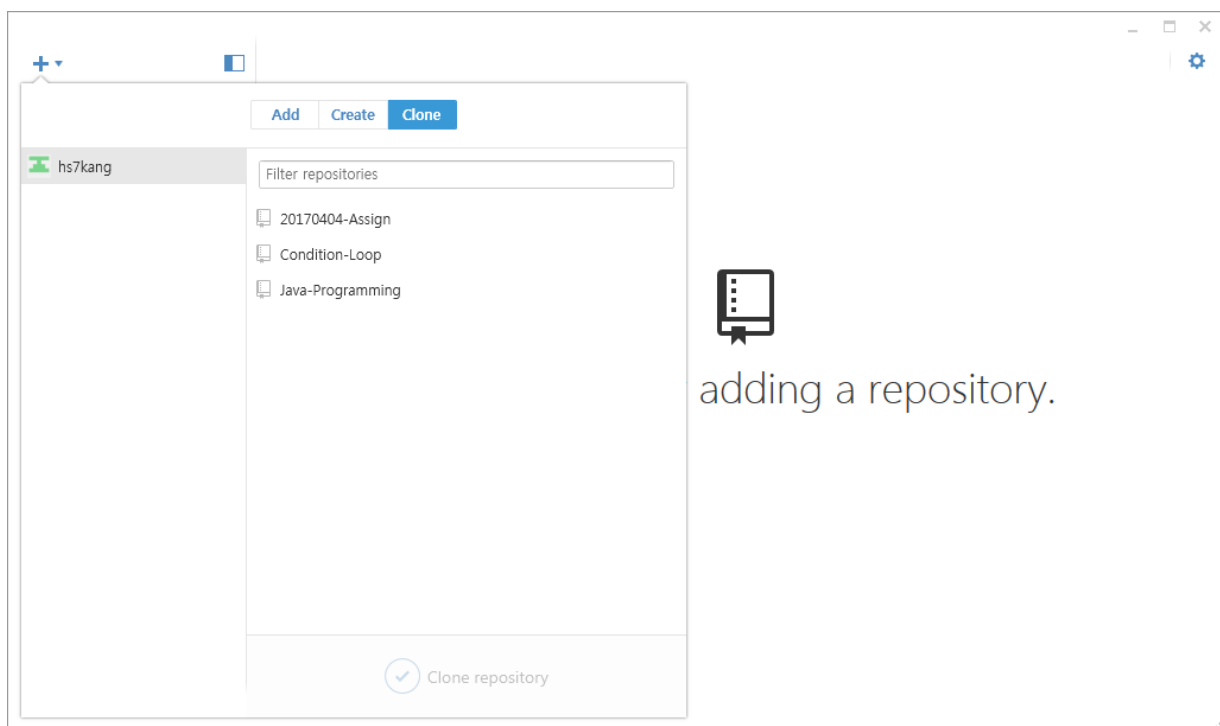
Privacy

☒ Help us improve by sending anonymous usage data

☒ Save ☐ Cancel

옵션 저장이 성공적이라면 우측 상단의 +를 눌러 **Clone** 을 선택해 보자. 다음과 같이 원격저장소인 자신의 깃허브 저장소 목록이 보인다. 생성자에 관련된 메뉴는 **Add, Create, Clone** 이 제공된다

- **Add:** 지역저장소를 기존의 깃허브 원격저장소에 추가
- **Create:** 지역저장소를 새로이 생성하여 깃허브 원격저장소도 함께 생성
- **Clone:** 깃허브의 원격저장소를 지역저장소로 복사



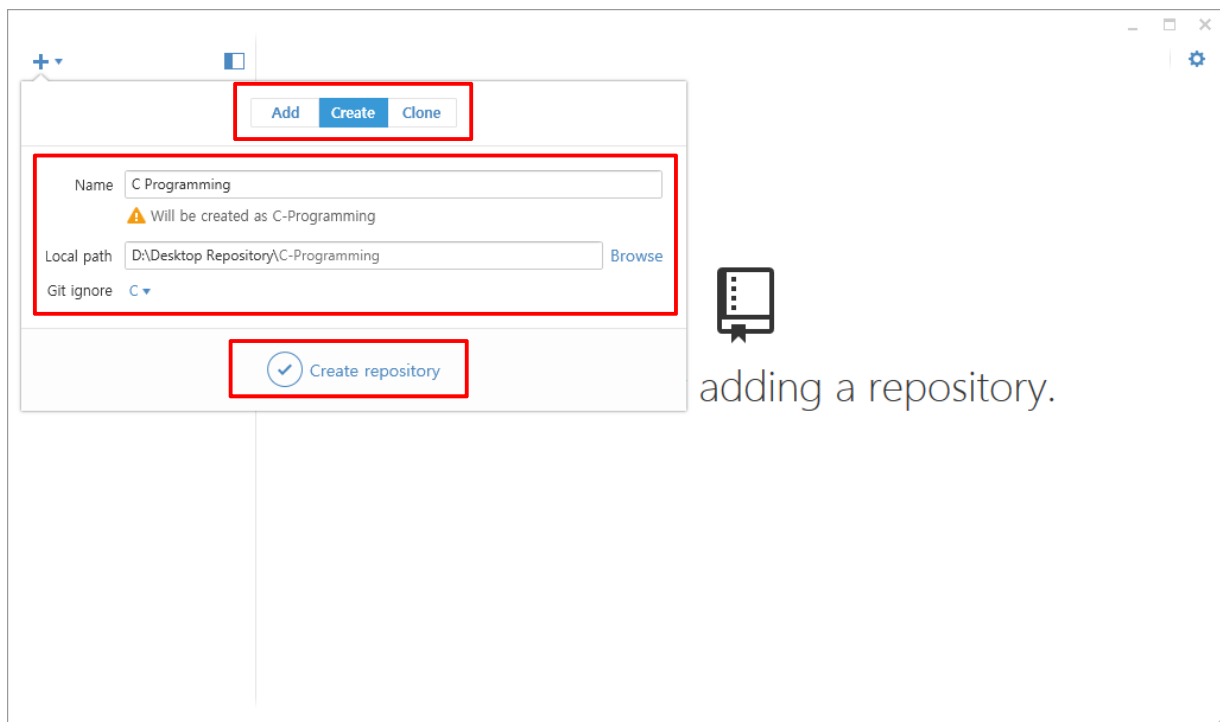
깃허브 데스크탑 지역저장소 생성과 업로드

깃허브 데스크탑 지역저장소 생성

이제 사용하는 자신의 PC 의 데스크탑에서 지역저장소를 하나 생성하자. 이 지역저장소로 지정된 폴더 하부는 모두 깃허브에 원격저장소를 만들어 업로드될 수 있다. 실행된 깃허브 데스크탑 좌측 상단의 +를 눌러 Create 를 선택해 Name 과 Local path 를 입력한다.

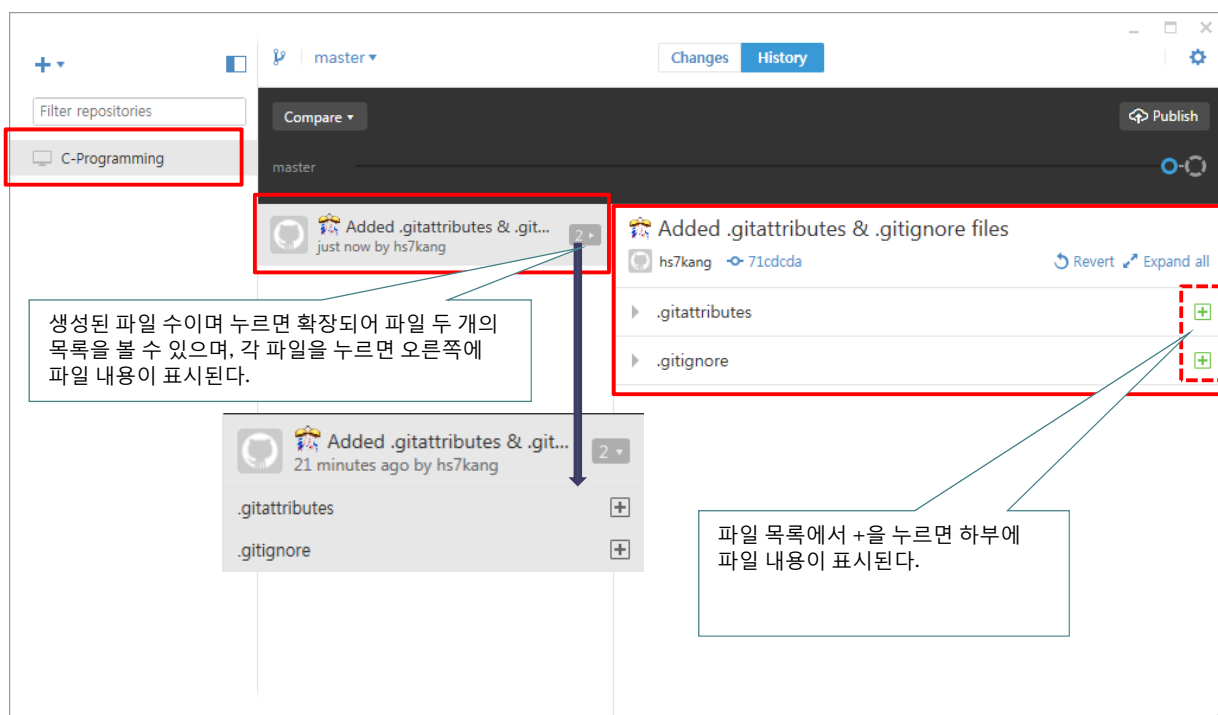
- **Name:** 지역저장소의 이름이며, 자동으로 깃허브 원격저장소의 이름으로 사용
- **Local path:** 실제 지역저장소가 저장될 상위 폴더
- **Git ignore:** 저장할 주요 소스 파일의 언어 이름을 선정하면 시스템에 필요한 설정파일 등을 자동으로 필터해서 업로드에서 제거

Local path 는 옵션에서 지정한 폴더가 자동으로 지정되며 필요하면 Browse 로 수정할 수 있다. Name 을 입력하면 지정된 Local path 하부 폴더로 자동으로 입력되는 것을 확인할 수 있는데 공백은 -로 자동 삽입된다. 입력이 완료되면 하단 Create repository 를 눌러 지역저장소를 생성한다.



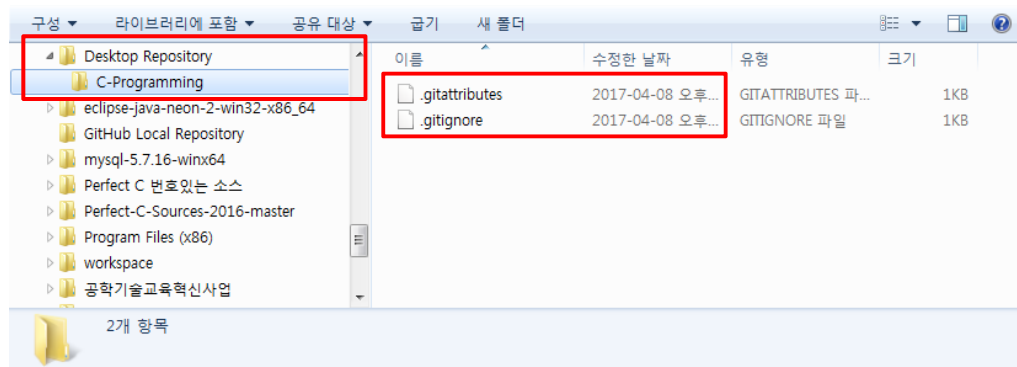
다음 화면은 저장소이름 입력 'C Programming'으로 역저장소가 생성된 모습이다. 처음 보는 낯선 파일 목록 2 개가 보이며, +를 누르면 그 내용도 볼 수 있다. 이 파일은 지역저장소를 관리하는데 사용되는 파일이다.

- .gitattributes:
- .gitignore:



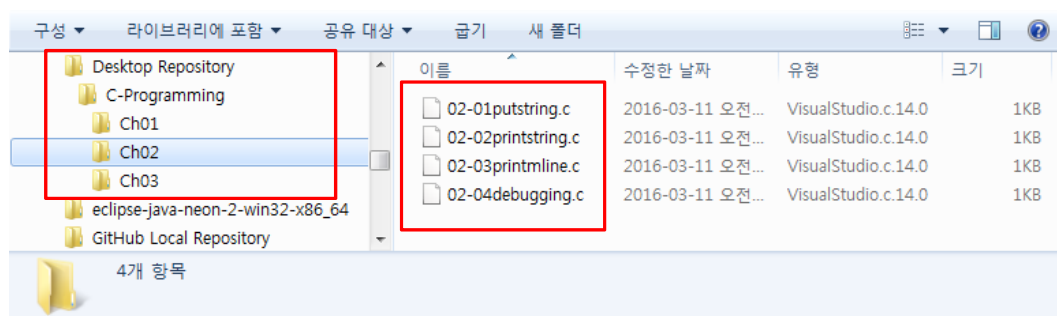
탐색기에서 생성된 실제 지역저장소인 폴더를 살펴보면 다음과 같이 초기화 파일 2 개가 생성된 것을 확인할 수 있다.

- .gitattributes
- .gitignore



지역저장소 구성과 깃허브 데스크탑 반영

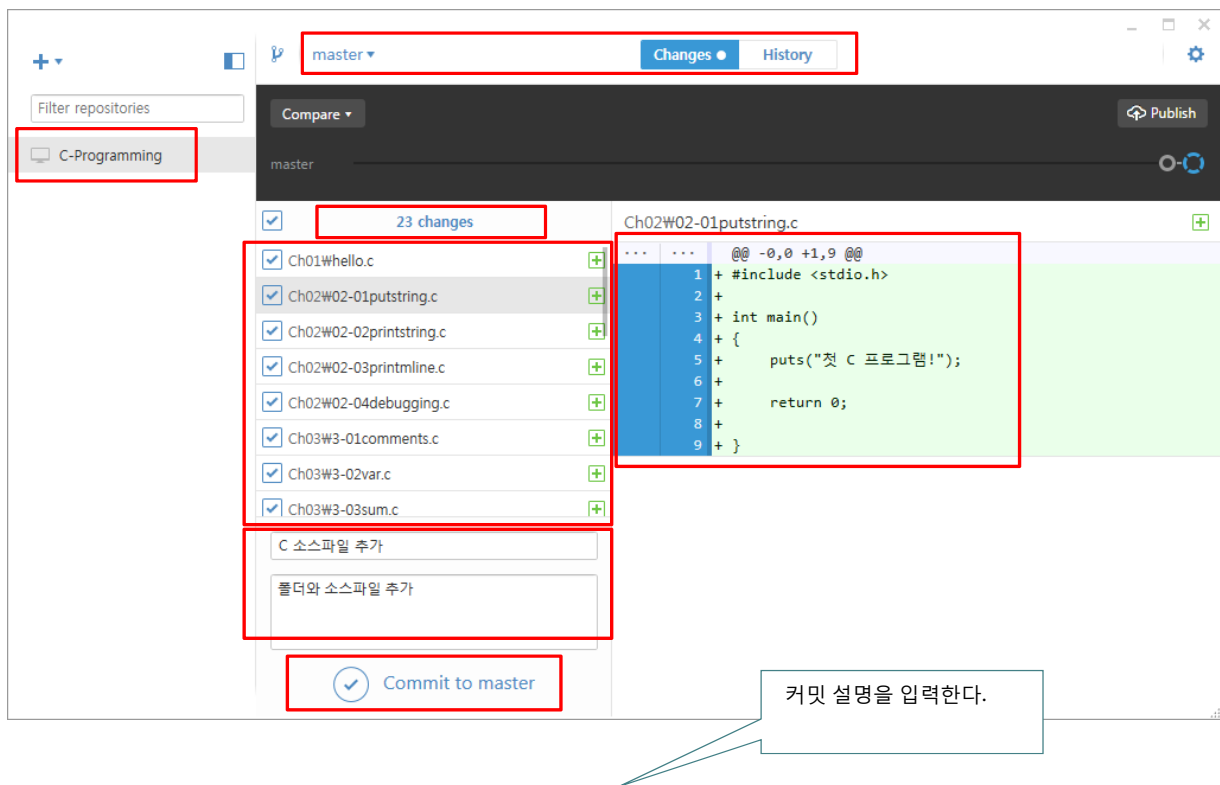
지역저장소 폴더 하부에 업로드할 폴더와 파일을 구성하자. 다음과 같이 폴더 **C-Programming** 하부에 여러 폴더와 파일을 생성할 수 있다.



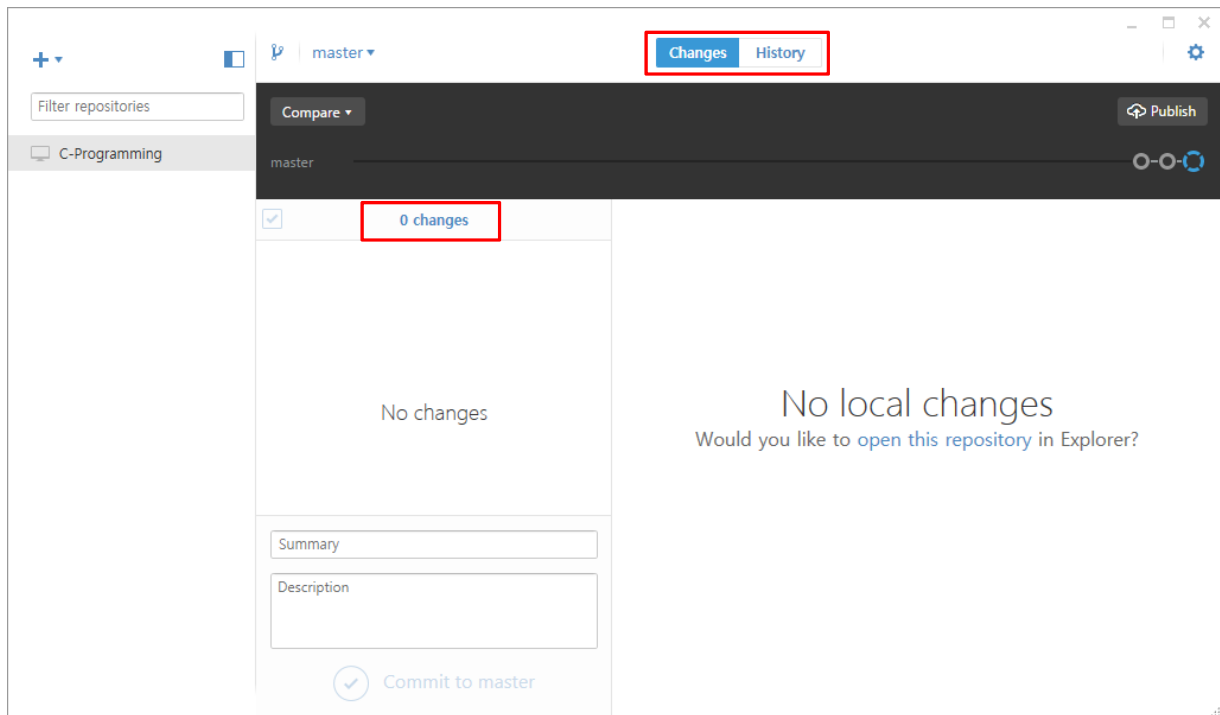
지역저장소 구성 이후, 실행된 깃허브 데스크 탑의 중앙 상단에서 **Changes** 를 누르면 다음과 같이 지역저장소가 수정된 내역 목록이 보인다. 이 작업을 통해 수정 파일을 지역저장소에 반영하는 커밋을 수행하게 된다.

- **23 Change:** 현재 지역저장소에서 23 개의 파일이 변화(changes)된 것을 표시
- 하부목록: 변화(changes)된 각 파일 목록을 표시되며, 각 파일을 누르면 우측에 파일 내용이 표시
- 파일 내용: 줄 번호가 표시되며 각 줄 앞의 +는 그 줄이 추가된 것을 의미
- **Summary:** 커밋 제목을 입력
- **Description:** 커밋 설명을 입력

- **Commit to master:** 수정된 내용을 지역저장소에 반영
- **커밋(commit):** 커밋을 실행한 시점의 저장소의 변화와 저장 내용을 저장

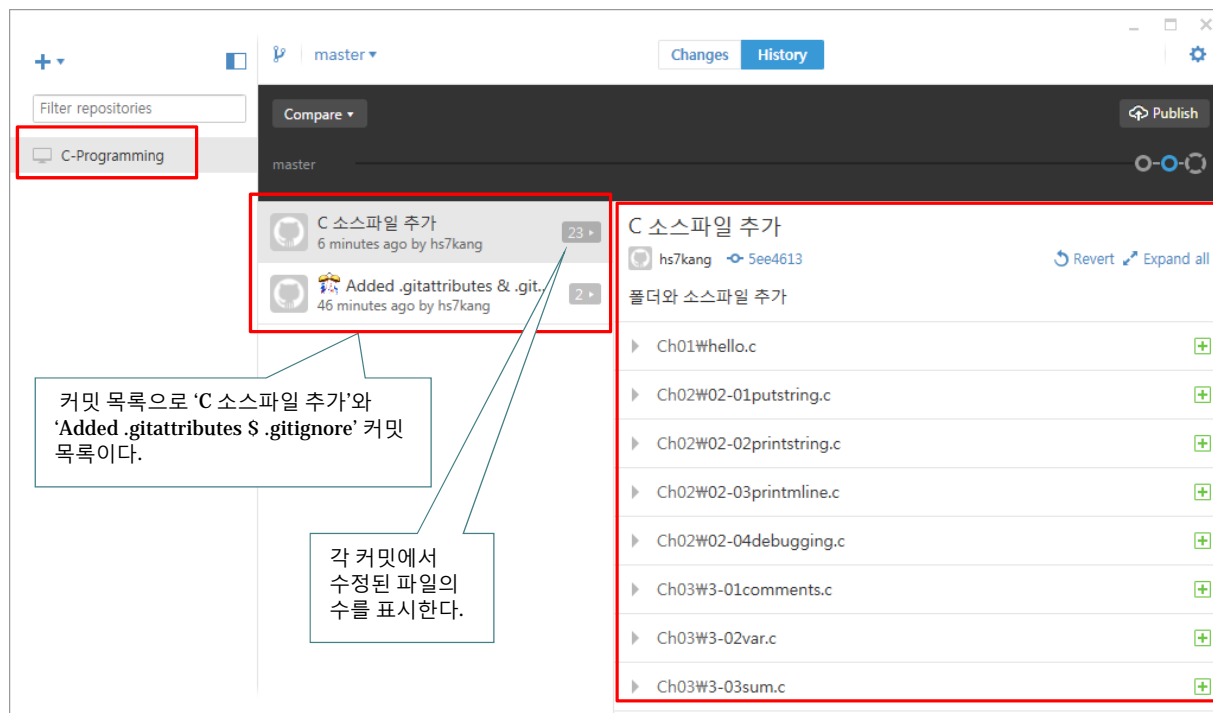


하부 Summary와 Description에 커밋 제목과 설명을 입력한 후 Commit to master를 누르면 수정된 내용이 저장소에 반영되는 커밋이 수행되며, 성공하면 이제 변화된 파일 없으므로 0 changes가 표시된다.



커밋 수행 이후 다시 중앙 상단의 **History** 를 누르면 커밋 제목인 'C 소스파일 추가' 제목으로 수정 내역을 확인할 수 있다. 중앙 목록의 각각을 커밋이라 한다. 다음 화면에서는 지역저장소 **C-Programming** 에 커밋이 2 개 있는 것을 알 수 있다. 화면의 왼쪽부터 각각의 열린 저장소 목록, 해당 커밋 목록, 해당 커밋에서 수정된 파일 목록이 표시된다.

- 저장소 목록
- 커밋 목록
- 커밋에서 수정된 파일 목록



제목이 'C 소스파일 추가'인 커밋은 추가된 23 개의 소스 파일의 추가에 대한 커밋이며, 'Added .gitattriburtes & .gitignore' 는 처음으로 지역저장소를 생성한 내용의 커밋임을 알 수 있다. 저장소를 관리하는 관점에서 언제나 커밋을 수행할 수 있다. 하나의 커밋을 선택하면 오른쪽에 커밋 내용이 표시되는데, 추가된 파일 `putstring.c` 의 +를 확장하면 다음과 같이 파일 내부 소스도 확인해 볼 수 있다. 가장 앞에 줄 번호가 표시되며 각 줄 앞의 +와 녹색 바탕 색은 그 줄이 추가된 것을 의미한다. 다음은 모든 줄이 추가되었으므로 바탕색이 모두 녹색임을 알 수 있다.



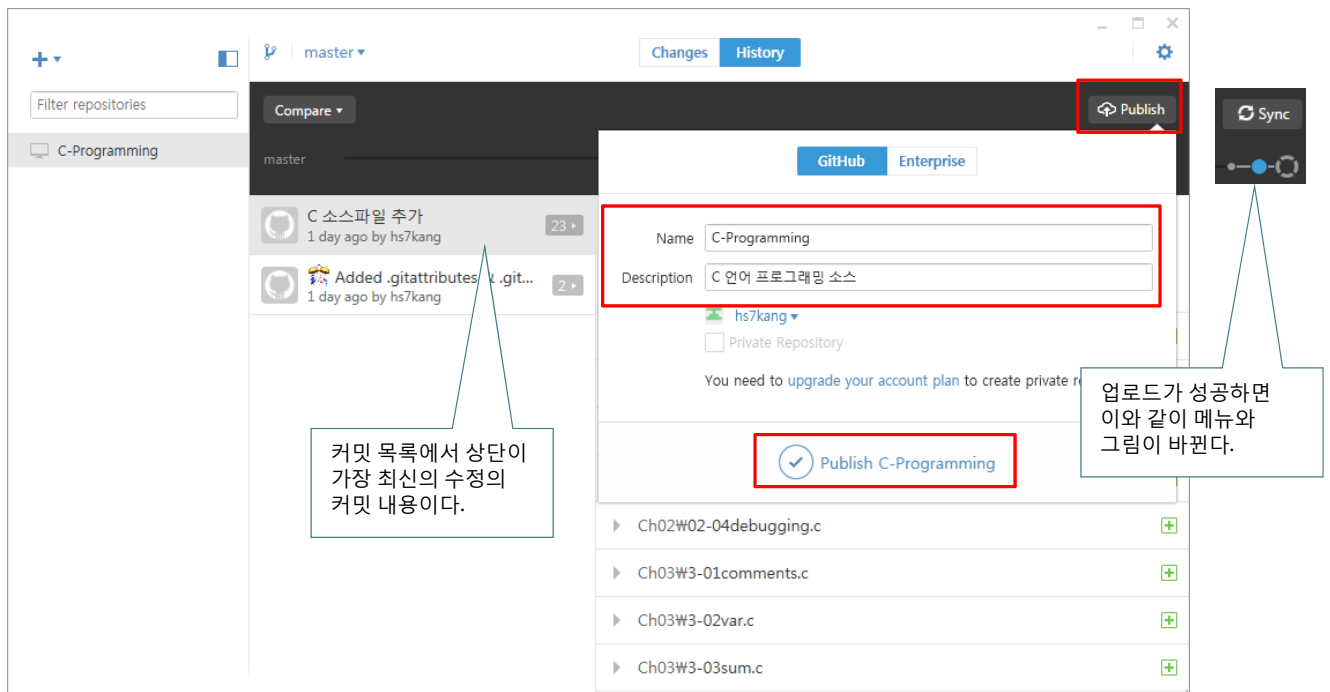
Tip

일반적으로 저장소에 중요한 수정이 발생하면 수행할 수 있으며, 파일을 편집할 때 중간 중간에 파일을 저장하는 것과 같다. 커밋을 저장소 전체를 중간 중간에 저장하는 개념으로 이해하면 좋다. 그렇다고 커밋이 시간과 공간을 많이 할애하지 않으므로 자주해도 성능에는 큰 영향을 미치지 않는다.

생성된 지역저장소를 깃허브에 업로드하자

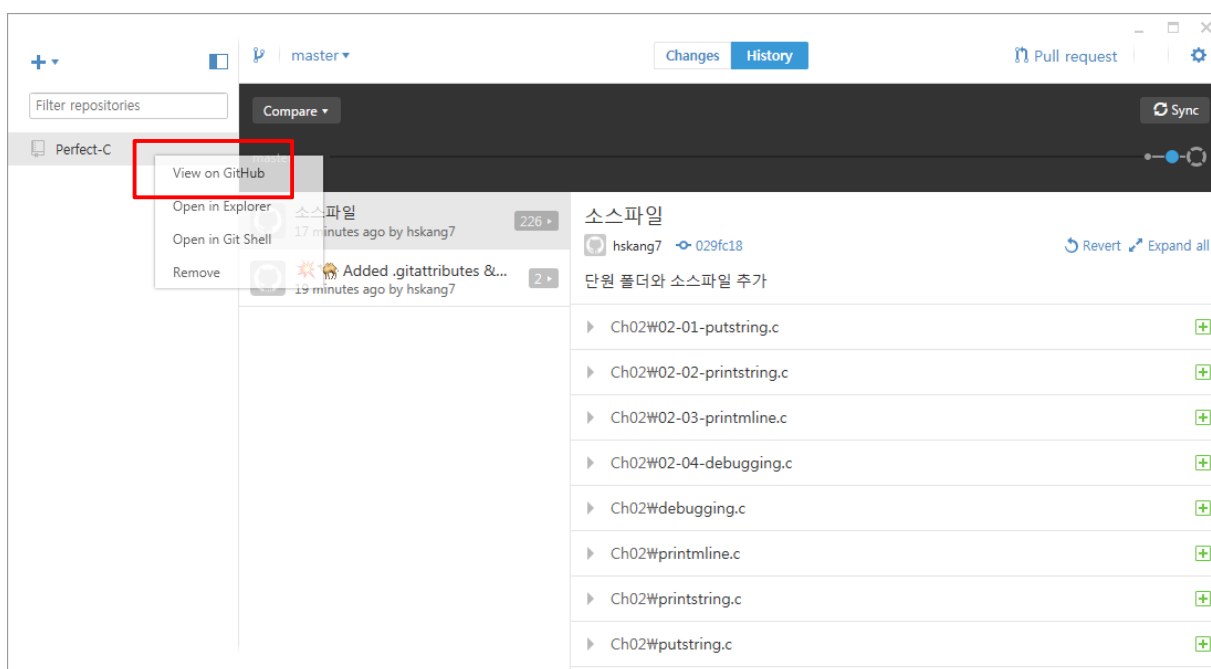
이제 지역저장소에 구성된 폴더와 파일을 웹 서비스하기 위해 깃허브에 업로드 하자. 지역저장소의 첫 업로드를 위해서는 저장소를 선택한 후 우측상단 **Publish** 를 선택한다. 표시된 대화상자에서 **Name** 과 **Description** 을 작성한 후 자신의 깃허브 ID 를 선택한 후 하단의 **Publish C-Programming** 을 누른다.

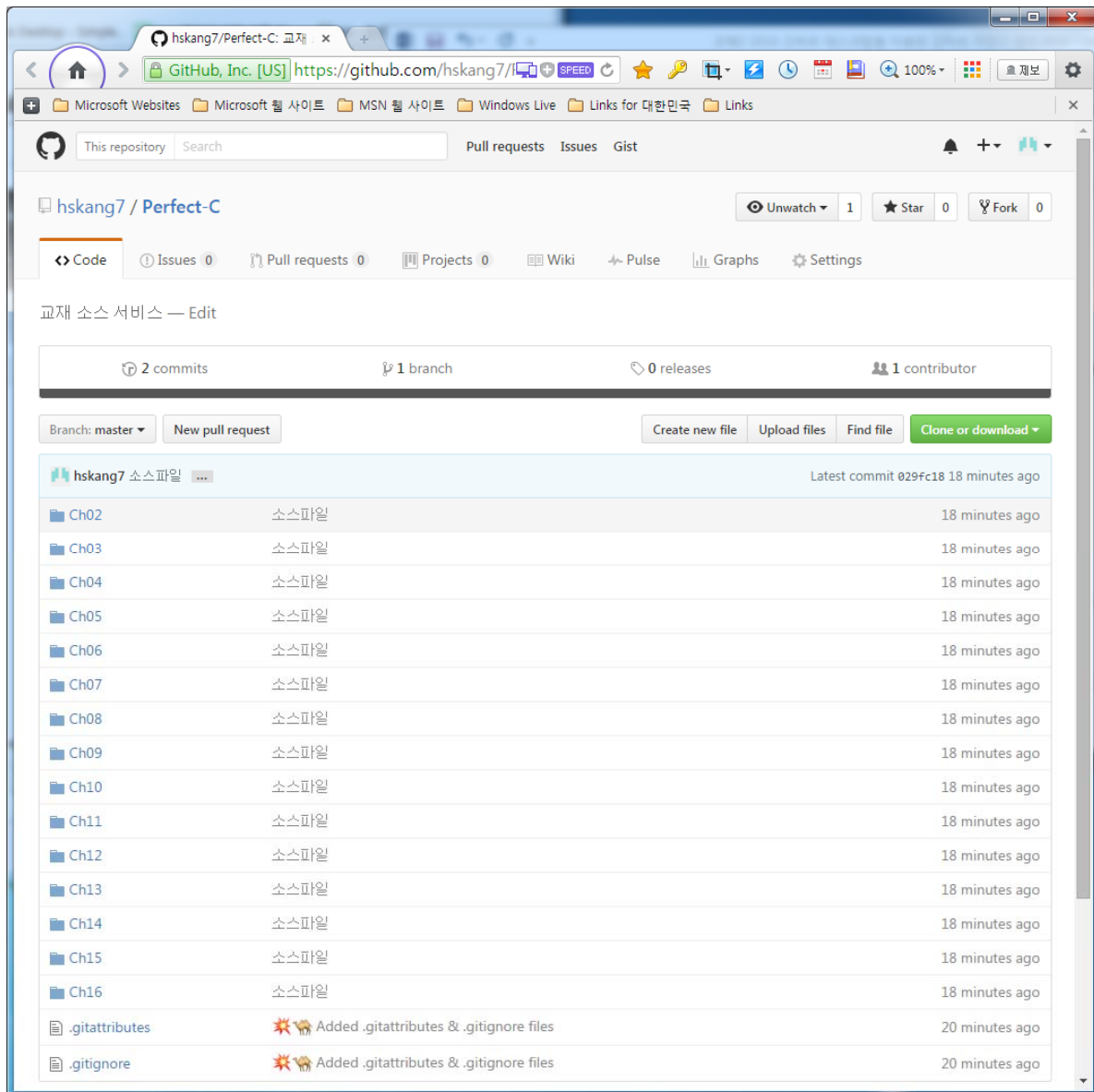
- **Publish:** 생성된 지역저장소를 처음으로 깃허브의 원격저장소를 만들어 업로드하는 메뉴
- **Name:** 깃허브의 원격저장소 이름으로 기본적으로 지역저장소 이름이 설정되며 원하면 수정 가능
- **Description:** 설명



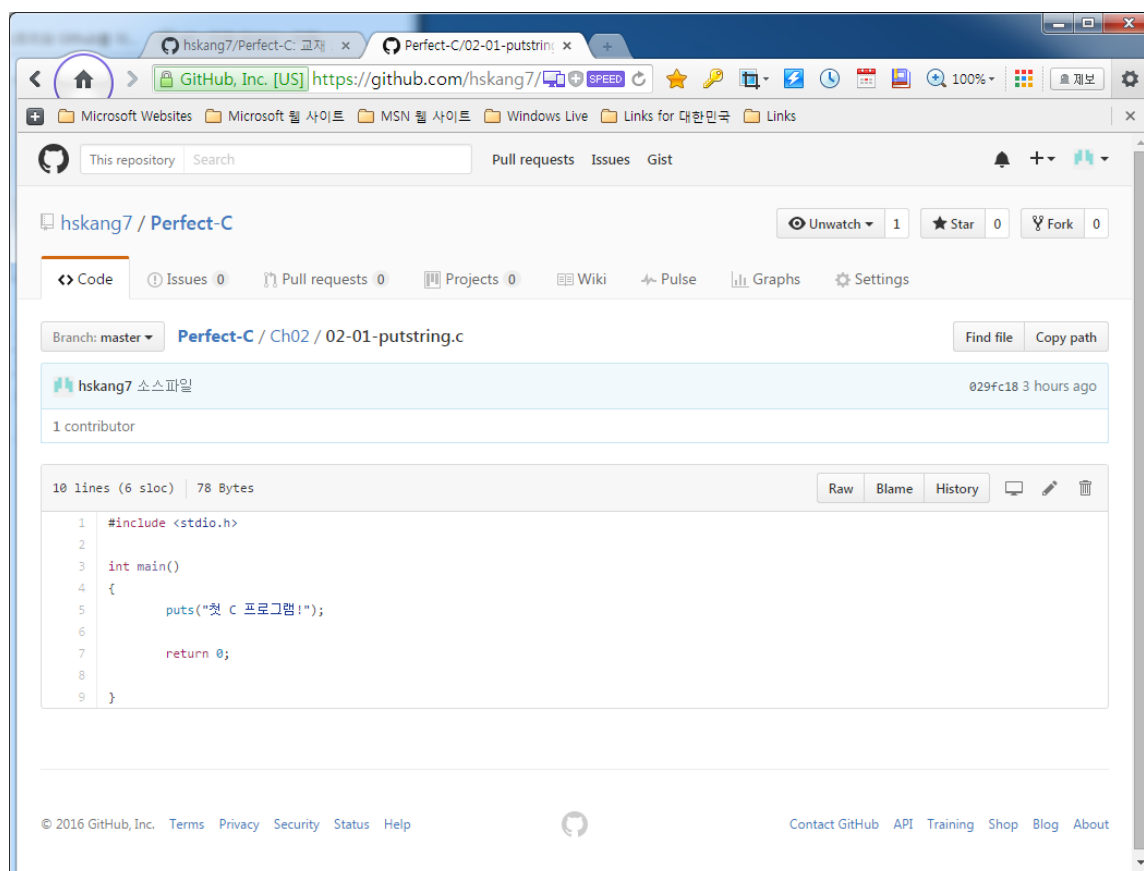
깃허브에서 업로드된 저장소를 직접 확인하기 위해서는 깃허브 데스크탑에서 지역저장소를 선택한 후 메뉴 **View on GitHub** 를 선택하면 편리하다. 물론 브라우저의 주소에 다음 저장소 주소를 입력해도 바로 깃허브의 원격저장소에 업로드된 내용을 확인할 수 있다. 이제 누구든 다음 주소로 업로드된 저장소의 파일 접근이 가능하다.

- <https://github.com/hskang7/Perfect-C>





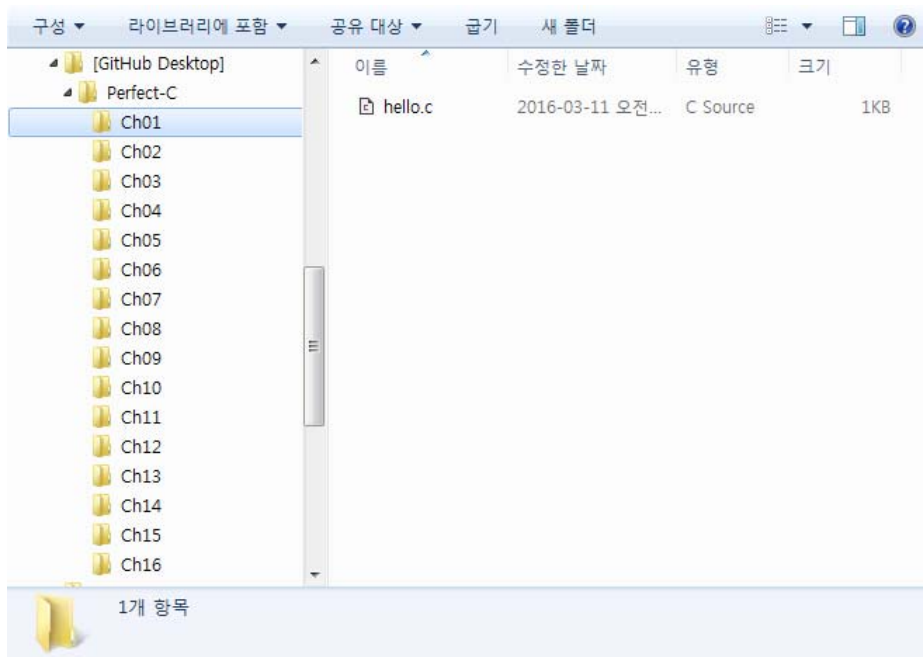
브라우저에서 원하는 폴더를 선택하여 파일을 하나 선택하면 다음과 같이 그 내용도 쉽게 볼 수 있다. C 소스파일이 키워드나 식별자의 색상을 구별하여 보기 좋게 표시되는 것을 볼 수 있다.



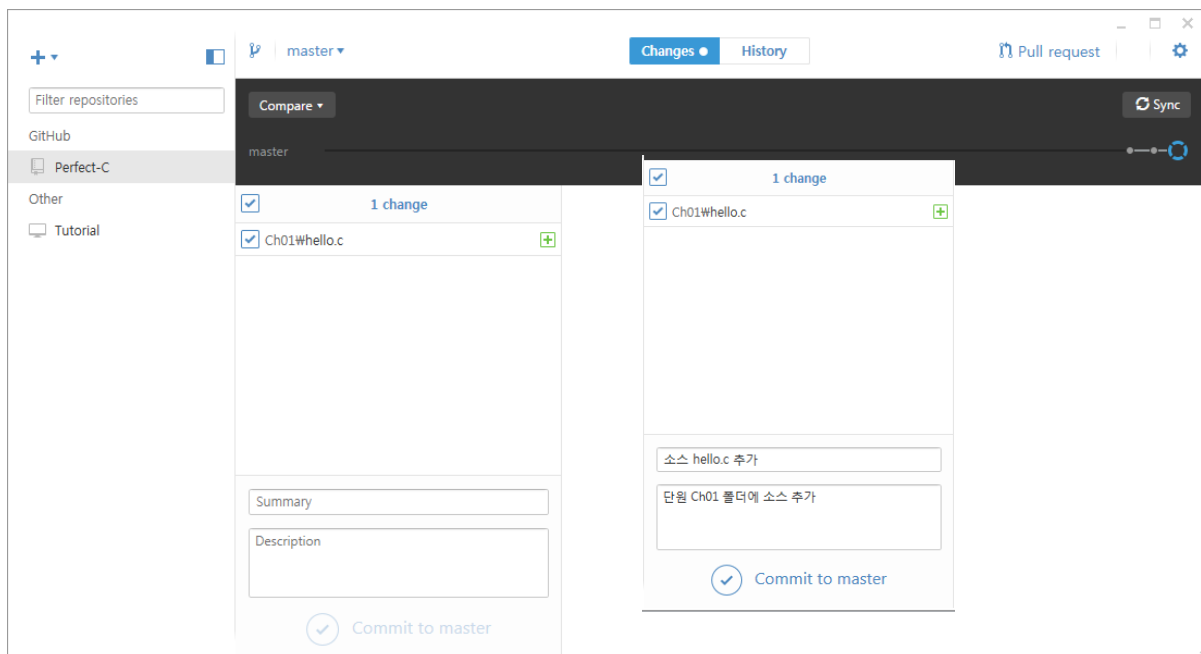
깃허브 데스크탑 지역저장소 커밋의 깃허브 반영

깃허브 데스크탑 지역저장소의 수정

만일 지역저장소에 다시 파일이 추가되거나 수정이 발생되면 이러한 지역저장소의 수정을 커밋하고 다시 깃허브 서버에 반영하도록 하자. 다음은 지역저장소 Perfect-C 에 폴더 Ch01 에 파일 hello.c 가 추가된 모습이다.

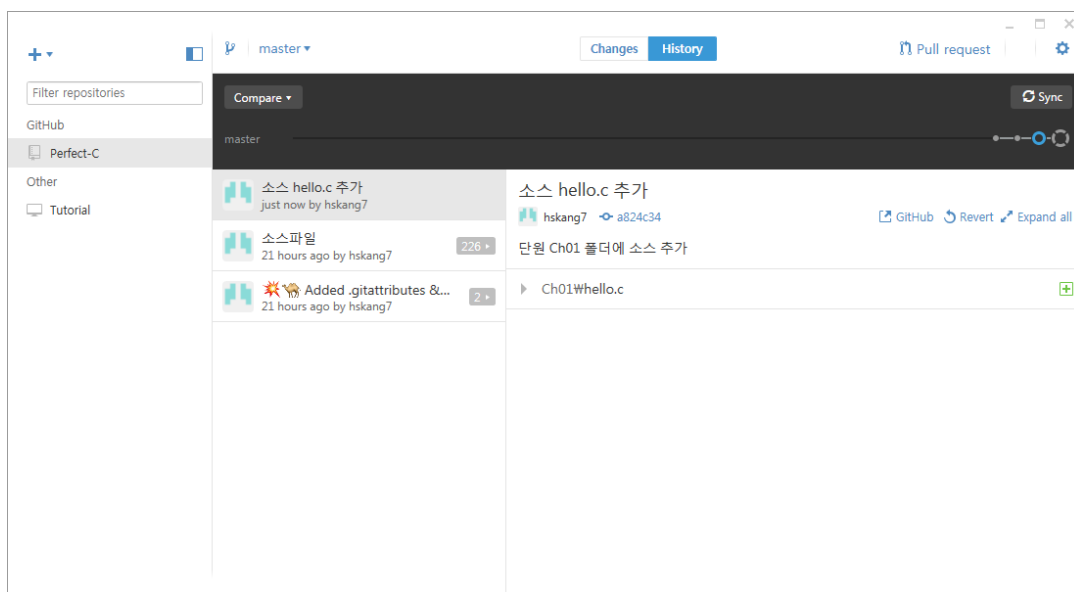


지역저장소의 수정이 발생한 이후에 깃허브 데스크탑에서 **Changes** 을 눌러 수정 내용을 살펴보면 다음과 1 **Changes** 가 표시되고 파일 **hello.c** 가 추가된 것을 확인할 수 있다.



깃허브에 수정 반영

먼저 위와 같이 지역정소의 수정 내용을 이름과 설명을 입력해 커밋을 먼저 수행하도록 한다. 이제 이러한 지역저장소의 변화를 깃허브 저장소에 반영하려면 깃허브 데스크탑의 우측상단 **Sync** 버튼을 누른다.



다시 브라우저로 깃허브 저장소를 살펴보면 다음과 같이 **hello.c** 가 추가된 것을 볼 수 있다.

