COURSEWARE

Introduction to Source Control

Contents

Overview

Source control is known by a few different names, such as **version control** and **revision control**. It is the practice of tracking and managing changes to code.

Source control is vital for managing software development projects. Being able to track changes to code allows developers to:

- centralise all code changes and additions to one code repository
- allow for simple and effective collaboration within development teams
- control the integration of new code into the codebase
- track changes from the entire team over the full lifetime of the project
- revert code back to previous versions

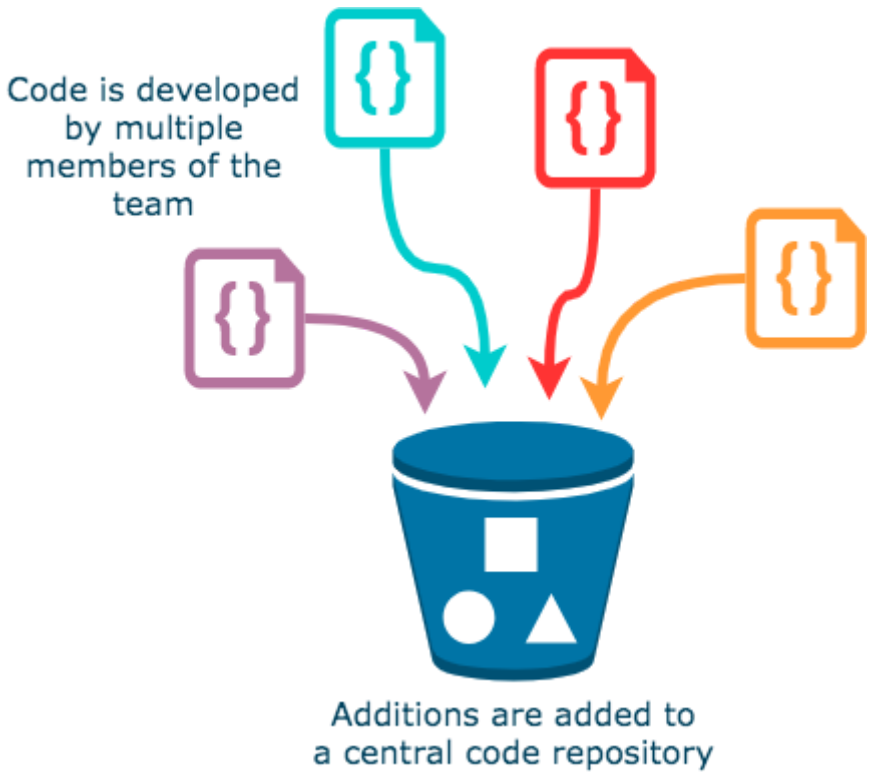
This module covers the benefits of using source control for software development, how we make use of source control using **source control management** systems, and finally some of the most common SCM tools that are out there.

Source Control Management (SCM)

We use *source control management* (SCM) systems to implement source control practices. These provide the ability to:

- Store code in a central repository
- Track changes over time
- Create code branches so that additions are made in isolation from stable code
- Merge new code into a stable release branch, known as the main branch
- Integrate with CI/CD automation tools (such as Jenkins and CircleCI) such that code will be built and tested as it is generated and pushed to the repository

Repositories

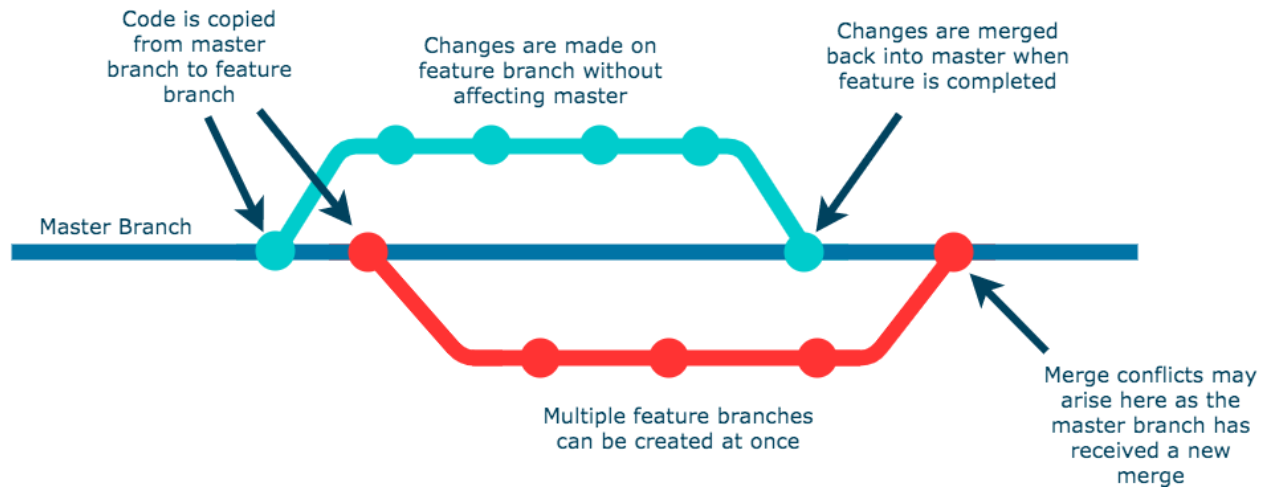


To keep code content in one place, SCM systems store code in a **repository**. This essentially functions as a big bucket for all your code to live in.

Having one place where your code resides means that versioning stays consistent, regardless of who's working on it.

This is vital when developers are working on the same project in tandem with one another – if there was no repository, there would be no cohesion each member's varying versions as they made changes to the code base.

Branching



Without source control, having multiple developers developing on the same code base simultaneously inevitably results in code conflicts and breakages to functionality, as two developers may need to work on the same parts of code to create their particular features.

Branching allows for each developer to work on new features in isolation by creating a version of the source code that only they are working on.

This new version is called a **feature branch**, while the source code lives on the **main branch**.

Once changes have been made and the feature implemented, the feature branch code is merged back into the source code on the main branch.

The SCM system will automatically detect the differences between the feature and main branches and alert the developer if there are any conflicts.

Conflicts may arise if changes have been made to the main branch after the feature branch has been created. The SCM will clearly show the changes that need to be made before merging can occur.

Code Tracking

Code tracking allows development teams to keep track of all changes made to a project over time. This allows for greater organisation as all additions to code are fully documented and attributed to the developer who made them.

Problems can then be solved with ease as code contributors can be consulted directly regarding their code; if a contributor is no longer a part of the team and can't be consulted directly, their changes are still fully documented allowing the team to track their contributions despite their absence.

If a feature causes a bug in the software after it is merged to main, code tracking makes it easy to revert the main branch to a previous stable state until fixes can be made.

SCM Tools

Some common SCM tools are:

- Git
- Mercurial
- Subversion (often abbreviated to SVN)
- CVS
- Perforce

Git is by far the most common SCM system used in software development. **Git** is a *free* and *open source* version control system.

Its ubiquity is largely due to how easy it is to learn and its tiny footprint, which is a result of being written in low-level C code.

It is a decentralised SCM tool, meaning its operations are largely performed on your local computer and requires no communication with an external server, resulting in very fast performance.

Repository Hosting Services

There are several web-based version control repository hosting services out there, most of which either utilise or are wholly based upon Git, such as:

- GitHub
- GitLab
- Bitbucket
- SourceForge
- Launchpad

Many cloud providers have their own code repository offerings which offer simple and powerful integration with other cloud services:

- AWS CodeCommit
- Azure Repos (as part of Azure DevOps)
- Google Cloud Source Repositories

Tutorial

There is no tutorial for this module.

Exercises

Spend some time exploring the differences between the web-based code repository services out there. Look up what they offer and write down the pros and cons of each (as best as you can understand them).