

# Let's Ship This Thing! 🚀

By: Gary Ewan Park



**Hello! I'm Scottish!**



MS  
Sucks

An orange heart graphic, drawn with thick, expressive brushstrokes, is positioned behind the text. The heart is slightly tilted and overlaps the letters of 'MS' and 'Sucks'.

# Slides

<https://gep13.me/ShipltSlides>



# Code

<https://gep13.me/ShipltCode>



# Agenda

- What is an issue?
- What is Semantic Versioning?
- What branching strategy should I use?
- What is GitVersion?
- What is GitReleaseManager?

# Q

What is an  
**issue?**

Demo

Q. What is an issue?

A

**“...it is the focal point for work undertaken on a particular task/bug/feature in a product release.**



Q

What is  
**Semantic Versioning?**

A

**“...a simple set of rules and requirements that dictate how version numbers are assigned and incremented. These rules are based on, but not necessarily limited to, pre-existing widespread common practices in use in both closed and open-source software.**

# The Rules

Given a version number MAJOR.MINOR.PATCH, increment the:

- MAJOR version when you make incompatible API changes
- MINOR version when you add functionality in a backwards-compatible manner
- PATCH version when you make backwards-compatible bug fixes

Additional labels for pre-release and build metadata are available as extensions to the MAJOR.MINOR.PATCH format.

# Examples

- 0.1.0
- 0.3.13
- 1.0.0
- 0.2.0-alpha.3
- 0.2.0-alpha.3+Branch.develop.Sha.e6eb071cd30974b80d7e237b85e7729a1d791e1e

Q

How do you know  
**when to bump**  
a version number?

# Tools

- [PublicApiGenerator](#)
- [Microsoft.CodeAnalysis.PublicApiAnalyzers](#)

Q. How do you know when to bump a version number?

A

**There is no quick answer** 🙄

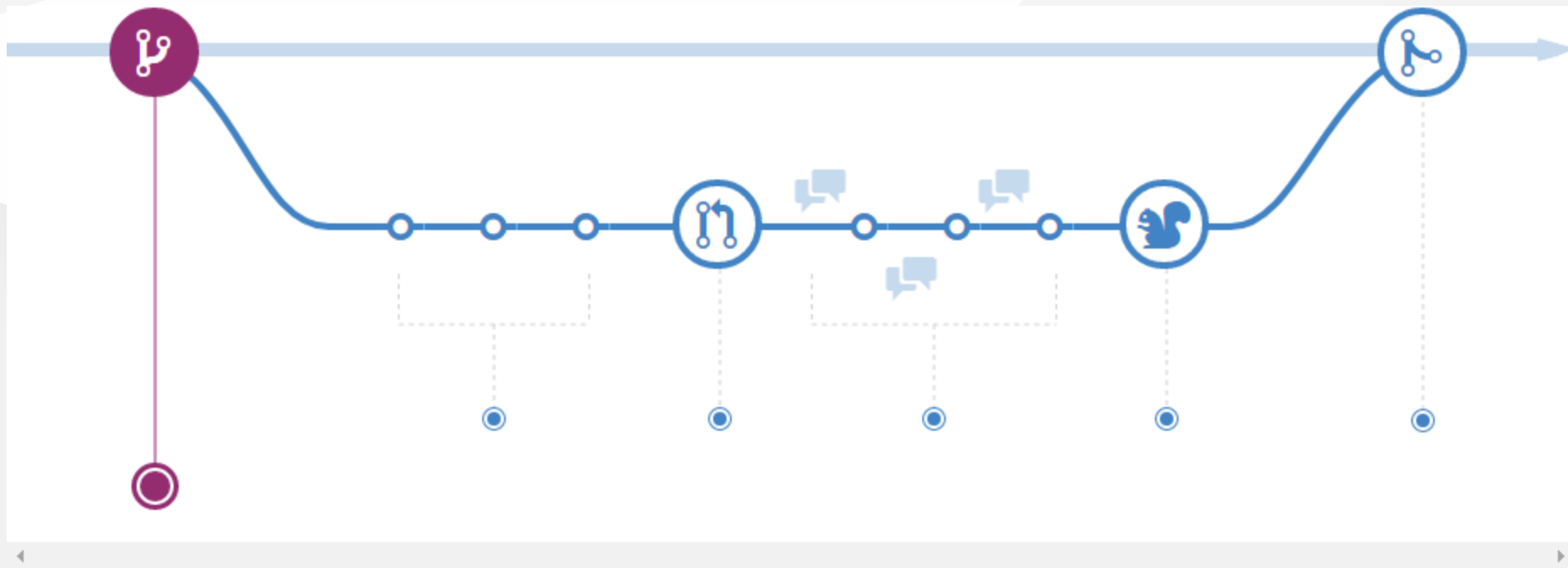
Q

What  
**branching strategy**  
should I use?

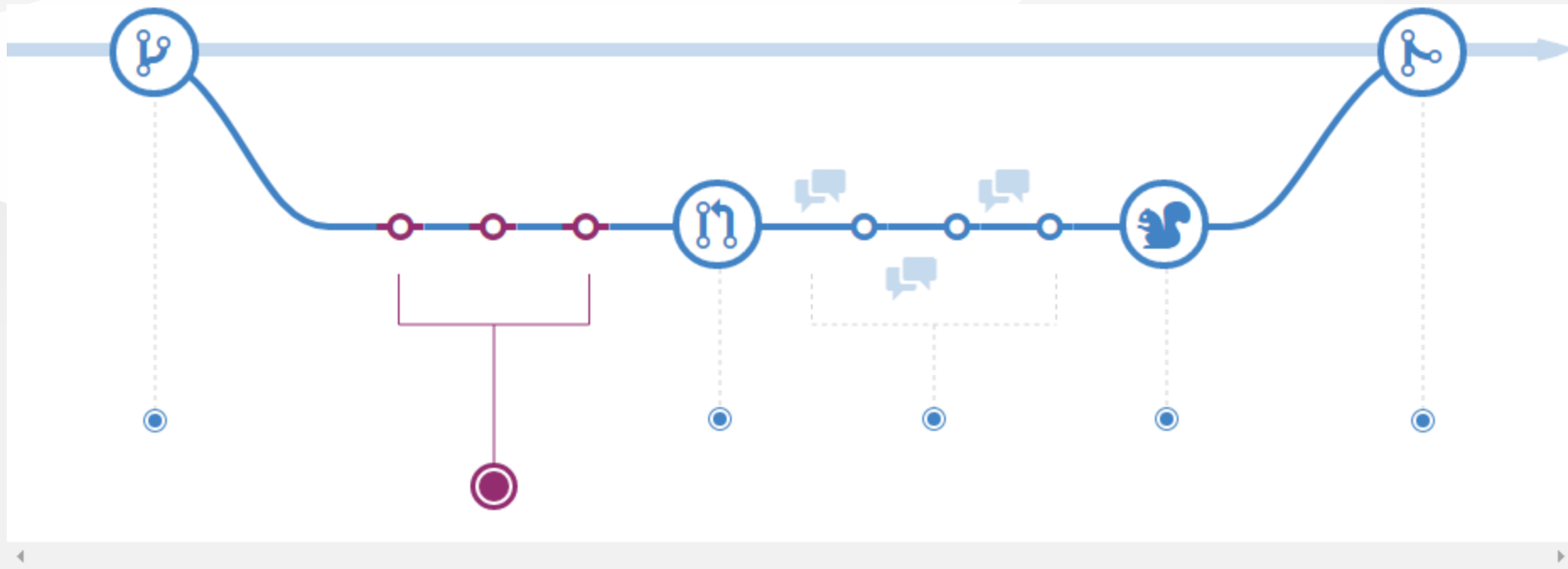


# GitHub Flow

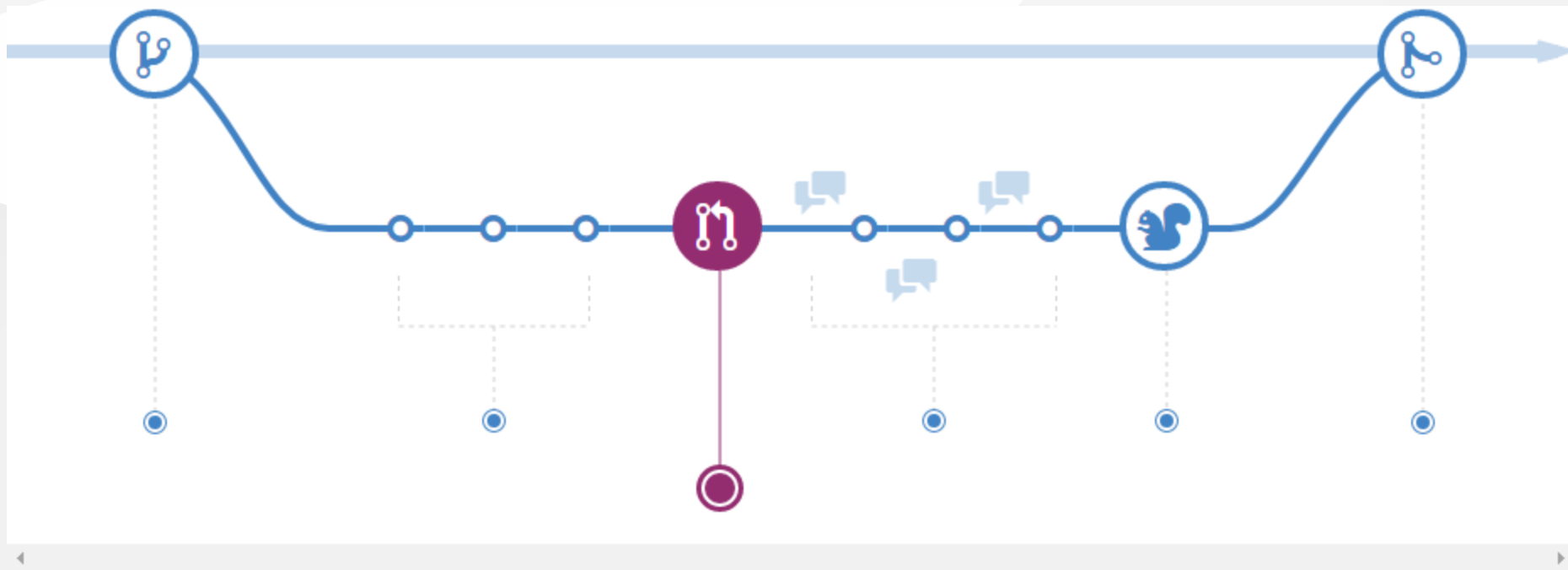
## Create a branch



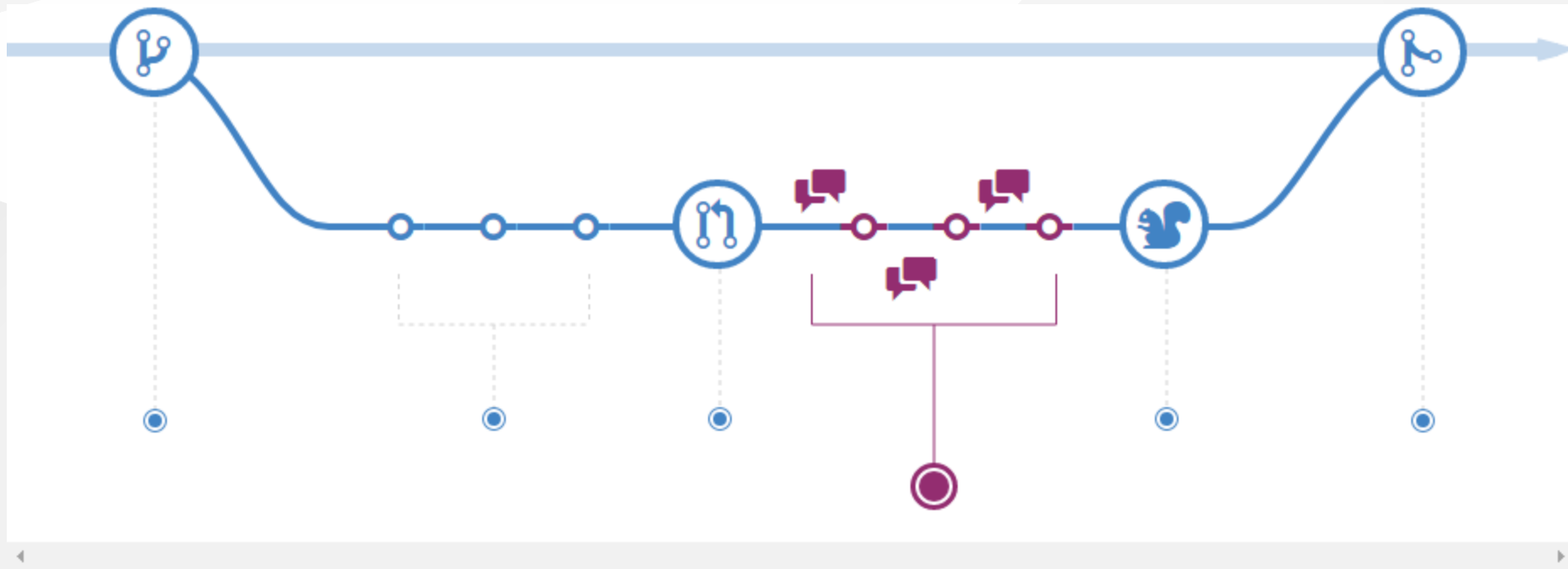
## Add commits



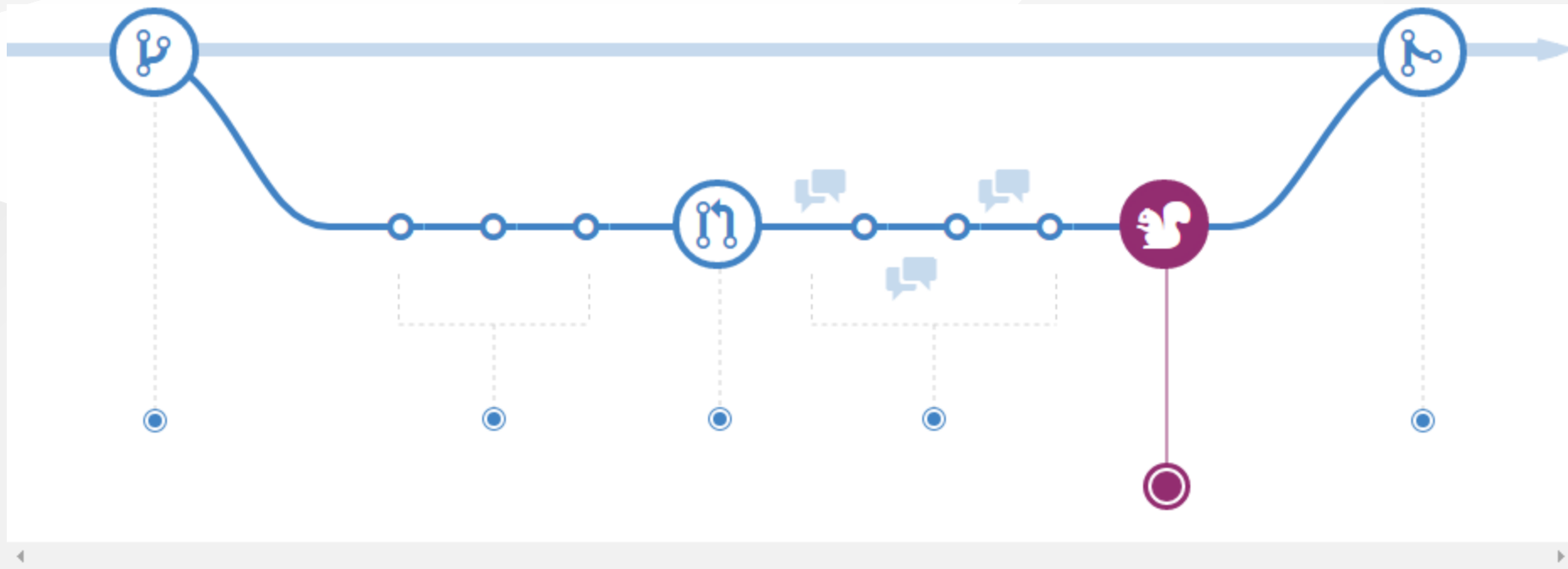
## Open a Pull Request



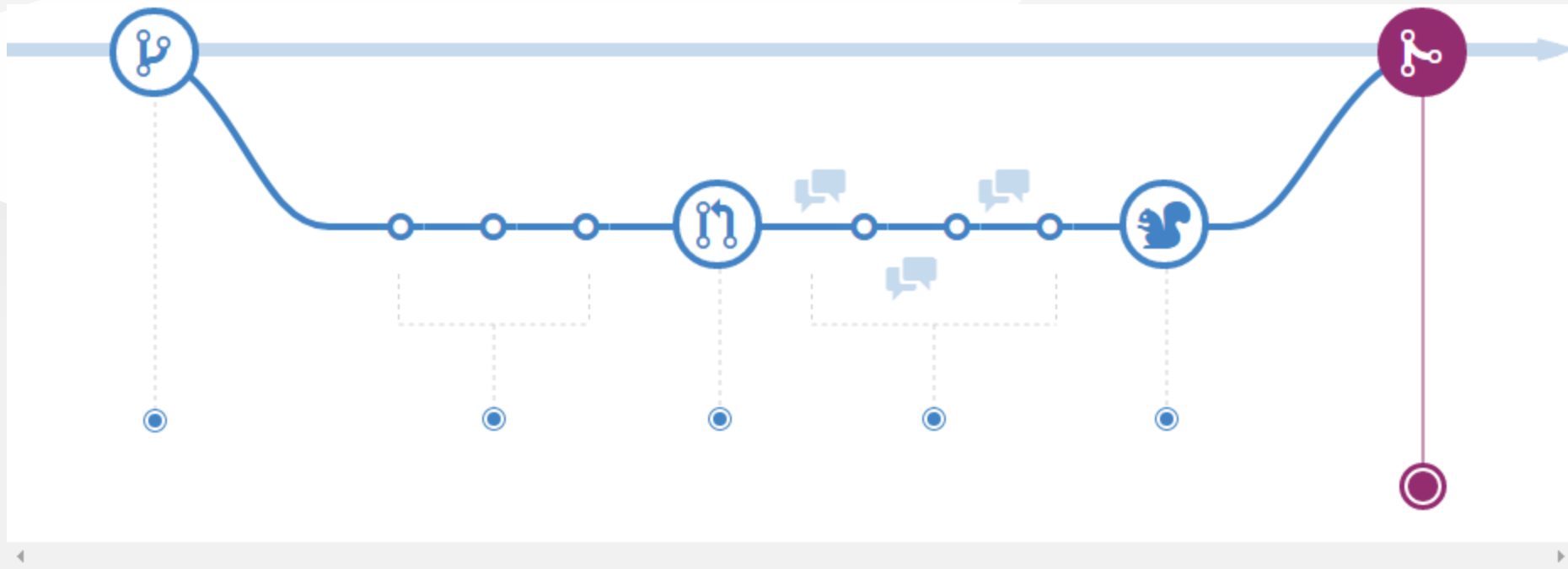
## Discuss and review your code



## Deploy

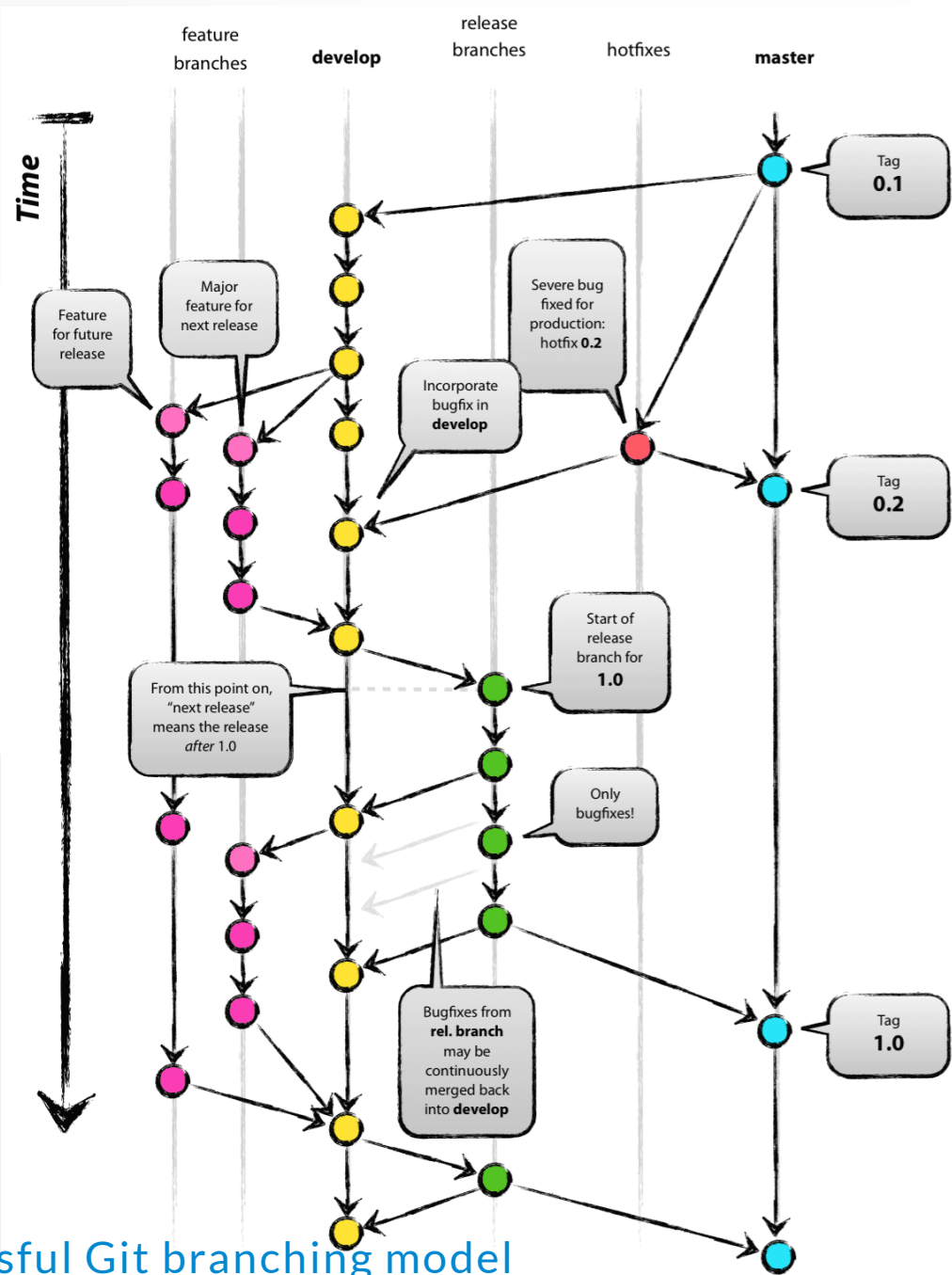


# Merge



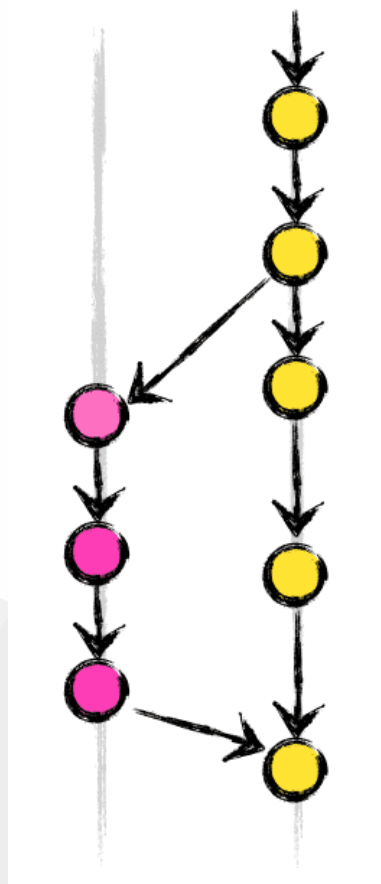
# GitFlow





Images borrowed from [A successful Git branching model](#)

feature  
branches      develop



Q. What branching strategy should I use?

A

**It depends!** 🧑

Q

What is  
**GitVersion?**

Q. What is GitVersion?
















A

**“...a tool to help you achieve  
Semantic Versioning on your project.**

Demo

Q

Why do I need  
**GitVersion?**

Name	Date modified	Type	Size
 .git	22/06/2015 11:13	File folder	
 BuildScripts	22/06/2015 11:13	File folder	
 Documentation	22/06/2015 11:13	File folder	
 Prototypes	22/06/2015 11:13	File folder	
 Source	17/07/2015 09:45	File folder	
 SQL Scripts	22/06/2015 11:13	File folder	
 Testing	22/06/2015 11:13	File folder	
 Tools	22/06/2015 11:13	File folder	
 .editorconfig	22/06/2015 11:13	EDITORCONFIG File	1 KB
 .gitattributes	22/06/2015 11:13	Text Document	1 KB
 .gitignore	22/06/2015 11:13	Text Document	3 KB
 build.bat	22/06/2015 11:13	Windows Batch File	1 KB
 generateHelp.bat	22/06/2015 11:13	Windows Batch File	1 KB
 README.md	22/06/2015 11:13	MD File	1 KB
 version.txt	02/09/2015 20:02	Text Document	0 KB



```
[assembly: AssemblyTrademark("")]  
[assembly: AssemblyCulture("")]  
[assembly: NeutralResourcesLanguage("en-GB")]  
  
[assembly: AssemblyVersion("0.2.0.0")]  
[assembly: AssemblyFileVersion("0.2.0.0")]  
[assembly: AssemblyInformationalVersion("0.2.0.0")]
```

Administration > <Root project> > CIDemo > Create Build Configuration

## General Settings

Name: <sup>\*</sup> Our First Build Configuration

Build configuration ID: <sup>?</sup> CIDemo\_OurFirstBuildConfiguration


This ID is used in URLs, REST API, HTTP requests to the server, and configuration settings in the TeamCity Data Directory.

Description: This will build and test our CIDemo Project

Build number format: <sup>\*</sup> <sup>?</sup> %build.counter% 

The format may include '%build.counter%' as a placeholder for the build counter value, for example, 1.%build.counter%.  
It may also contain a reference to any other available parameter, for example, %build.vcs.number.VCSRootName%.  
Note: The maximum length of a build number after all substitutions is 256 characters.

Build counter: <sup>\*</sup> <sup>?</sup> 1 [Reset counter](#)

Artifact paths: <sup>?</sup> Edit artifact paths:  


Newline- or comma-separated paths to build artifacts. Ant-style wildcards like `dir/**/*.zip` and target directories like `*.zip => winFiles, unix/distro.tgz => linuxFiles`, where `winFiles` and `linuxFiles` are target directories are supported.

Build options: <sup>?</sup> ☒ enable hanging builds detection  
☐ enable status widget <sup>?</sup>

Limit the number of simultaneously running builds (0 — unlimited) 0

[VCS settings >>](#)

[Cancel](#)



Q. Why do I need GitVersion?

A

**To make version assertion  
reliable and consistent**

Q

What is  
**GitReleaseManager?**

Q. What is GitReleaseManager?

A





**“...a tool that will help create, and manage, a release for your application/product.**

Demo



# Gary Ewan Park

**Principal Software Engineer**  
**Chocolatey Software, Inc**

-  Twitter: [@gep13](https://twitter.com/gep13)
-  Bluesky: [@gep13.co.uk](https://bsky.app/profile/gep13.co.uk)
-  Blog: <https://gep13.co.uk>
-  GitHub: [gep13](https://github.com/gep13)



# Questions

# Learn More

- GitVersion Documentation
  - <https://gitversion.net/docs/>
- GitReleaseManager Documentation
  - <https://gittools.github.io/GitReleaseManager/docs/>
- Git Branching Strategies
  - <https://www.atlassian.com/git/tutorials/comparing-workflows>
- GitFlow
  - <https://nvie.com/posts/a-successful-git-branching-model/>