7

Configuring Team City For CICD with Xamarin

In the old days, application development used to happen in separate not so integrated teams and developers were not used to merge their work with other developers’ code for quite long time and that used to create merge issues. Things that were working once on the developer’s local copy, used to stop working when merged with others’ code. This kind of not so integrated development environment increases the development time and delayed discovery of issues. Solution for this is to have continuous integration build into the development cycle where developers are merging their code multiple times a day and getting issues fixed at earlier stages.  
In this chapter we will learn more about Continuous Integration and Continuous Delivery and different tools that we can use for the same for a better development integration and delivery process.

**Introduction To Continuous Integration**

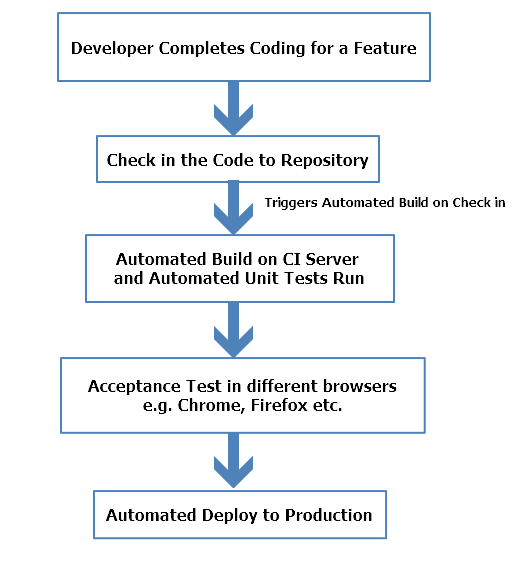
**Continuous Integration** (CI) is a development and integration practice in which developers checkin code into a shared repository frequently, preferably several times a day. Each code merge can then be verified by an automated build and automated tests if applicable.

There are many benefits of following continuous integration, one of the advantage is that it helps detect defects quickly and at early stage. The check ins are usually very small and contains small portions of developments, thus helping in identifying the exact issues quickly.

**Continuous Delivery** (CD)in the other hand is the process after integration and as the name suggests, it is to make sure that the code base checked in is deployable at any point of time. Each environment from test to production can and mostly do have different configuration. Continuous Delivery makes sure that all configurations are always ready for the deployment to any environment and that the code passes all the tests necessary for release.

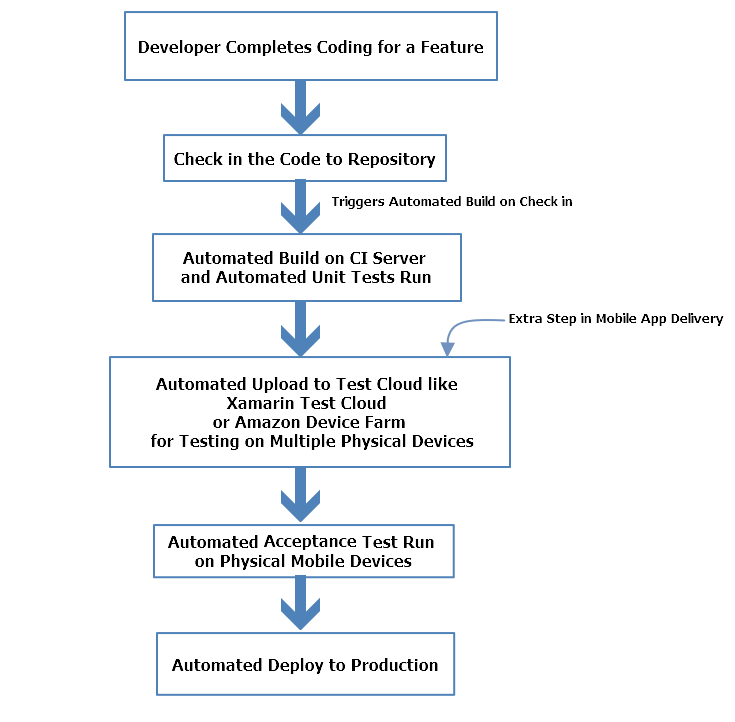
In short, Continuous integration improves the development and testing experience with frequent code merges and helps in quickly identifying the bugs and involves running automated tests if included in the process while Continuous delivery makes sure the codebase everything is in ready state for the code to be deployed in any environment.

**CICD for a Web Application**



For a web application once the build is ready from the CI server it is not a big task to test the application on different browsers since there are only limited number of them. But in a mobile application there is one more step involved to improve the experience since there are thousands of devices with different versions of operating systems available.

**CICD for a Mobile Application**



As shown in the image above, mobile applications need to be tested on hundreds of devices with different operating systems and purchasing on those mobile devices which keep on coming every now and then, can be very expensive. To make sure the quality of application stays at the top including Test Cloud based solutions becomes an integral part of the process.

**Choosing Tools for Continuous Integration**

There are many CI tools available in the market to implement continuous integration, just like there are many languages available in the market to developer applications. But choosing the right CI tool is very important for a good long-term benefit.

Choosing a CI tool for your project can depend on many variables like:

* **Programming Language Support**

This is one of the most important factors while choosing a CI tool. Some CI tools have better support for certain language specific builds and packages while others might not provide language specific packaging options.

* **Operating System**

Operating system support, meaning some teams might find an open source operating system like Linux to be a better choice for all their servers including the CI server and it might be more comfortable for them to configure a familiar operating system while other teams working on .Net applications might find Windows to be more comfortable and feature rich for there configurations. It all boils down to the preference different teams have and the kind of application they are working on.

* **Integration with Code Repository**

Different teams prefer different code repositories for various reasons, some might find Git to be more feature rich and supported on various IDEs with plugins while others who are more familiar with Microsoft environments find Team Services to be easier to use and better integrated. Different CI tools have different level of support for these repositories.

* **Support for Application Platform Deployment**

Some CI servers are better suited for web application deployment while other provide more features and better support for mobile application deployment to app stores. Depending on your type of application, the choice can vary.

* **Cost**

Cost is always an important factor while choosing any type of tool. Medium to big size companies can afford to have expensive feature rich CI tools while smaller companies and teams might want to stick to low budget and sometimes open source and freely available CI tools and customize them according to their needs.

**Various Tools for Continuous Integration**

Let’s have a look at some of the widely used CI tools available in the market.

**TeamCity**

TeamCity is a mature CI server, developed by JetBrains company. JetBrains is very popular in the software development world, and their tools like WebStorm and ReSharper are used by developers worldwide. It is a license based CI server but a free version of it is also available. TeamCity offers all the features in its free version, but it is limited to the 20 configurations and 3 build agents. Additional build agents and build configurations need to be purchased.

Despite being Java based solution, TeamCity **offers the best .NET support** among the tools on this list. There are also different enterprise packages, that scale by the number of agents needed.

You will be learning more about TeamCity later in the chapter.

**Key Features:**

* Great support for Visual Studio; versioning, testing, code coverage, code analysis, all available without any external scripts
* Detailed history reports for builds, failures, and any additional changes made
* TeamCity takes advantage of cloud computing by dynamically scaling out its build agents farm on Amazon EC2, Microsoft Azure, and VMware vSphere

**Jenkins**

Jenkins is one of the most popular open-source project for continuous integration. With thousands of plugins to choose from, Jenkins can help teams to automate any task that would otherwise put a time-consuming strain on your software team. Common uses include building projects, running tests, bug detection, code analysis, and project deployment.

**Key Features:**

* Jenkins works as a standalone CI server, or you can use it as a continuous delivery platform as well
* Pre-built packages for Unix, Windows, and OS X ensures an easy installation process
* A user-friendly web interface that can be used to quickly configure your server
* Custom plugins for build and source code management, administrative tasks, user interface, and platforms
* Large community with leading software brands involved in development

**Visual Studio Team Service**

Visual Studio Team Services provided by Microsoft helps teams plan better, code together, and ship faster. You can code in any IDE and language, for any target platform. Various tools and plugins can be downloaded to customize it to your project requirements.

**Key Features:**

* Kanban and scrum boards available for better planning and project management
* Unlimited Git and TFVC repos
* Hosted builds
* Automated release pipelines for better release planning
* Test and build commits before merging code to avoid build fails
* Built-in tasks and templates for setting up CI and CD to an Azure web app

**Bamboo**

Bamboo is a CI server being used by software teams worldwide to automate the process of release management for applications and general software, it allows teams to establish a streamlined pipeline of build delivery. Mobile developers can deploy their apps back to the Apple Store or Google Play automatically. Being the Atlassian tool, it has the native support for JIRA and BitBucket and you can even import your Jenkins configurations into the Bamboo easily.

**Key Features:**

* Bamboo can be used with Docker, AWS, and S3; it works out of the box with all of your favorite coding language
* Custom deployment projects to archive the history of each of your release version
* Detailed outline of your code history before you deploy, helping you understand the progress you’re making
* Compatible with Bitbucket and JIRA for a comprehensive CI experience
* With per-environment permissions, developers and QA can deploy to their own environments on demand while production stays locked down

**Requirements for using TeamCity with Xamarin**