1. What is continuous integration?

Checking every change in the source code as soon as possible, or as customized by the coder, and running the code to see the result.

1. Why to use continuous integration?
2. What was done before continuous integration was known?

Suppose a team of developers are working on a project. The developers would test it individually and store their codes in a repository (collection of all the source codes version-wise). There is a possibility that when the final application runs, it fails. This means the code broke somewhere. There can be multiple reasons for this:

* Two developers making changes on the same code at the same time.
* Developer makes a minor change and doesn’t test it again.
* A developer changes the class name but forgets to notify others.

When the final application breaks, it leads to “it was working on my system” dispute!

1. Advantages of continuous integration:-

Continuous development provides a great workflow because it automates testing.

-The codes will run continuously and would be tested for every change.

-You know where the error is as soon as you make a change.

3) The problem discussed above is not the only problem. Codes are useless if they don’t run somewhere. External dependencies create a problem like

i) Different server packages

ii) Different operating systems, etc

all

All these changes break the application. This is where continuous deployment is useful. It would be great to have all the source codes on one environment.

To achieve continuous deployment we need an infrastructure that automates different steps needed to deploy an application.

All the following steps are automated.

Getting Started

1. First of all we need to implement continuous integration. For that we need a centralized place where all the tests can be done on continuous interval. This place is called continuous integration server.
2. Hudson/Jenkins is a continuous integration tool.
3. Continuous deployment stages the data. Meaning stores the data from multiple source systems. It acts like a pool in which data from different source systems are stored for further processing.
4. Next step is the source control commit check. What it does is when something is going wrong at any point this script should halt any new commit. It should block any new entry to the repository.
5. Then we require real-time alerting. A monitoring platform that notifies where the things have gone wrong. It should notify the developer through mails, alerts, etc.

Requirements:

1. Source code repository
2. Continuous integration server
3. Build tools
4. Automation tools
5. Configuration
6. Test requirements

Build Tools

Building a program means constructing it. It is converting the source code to an application which can show some result.

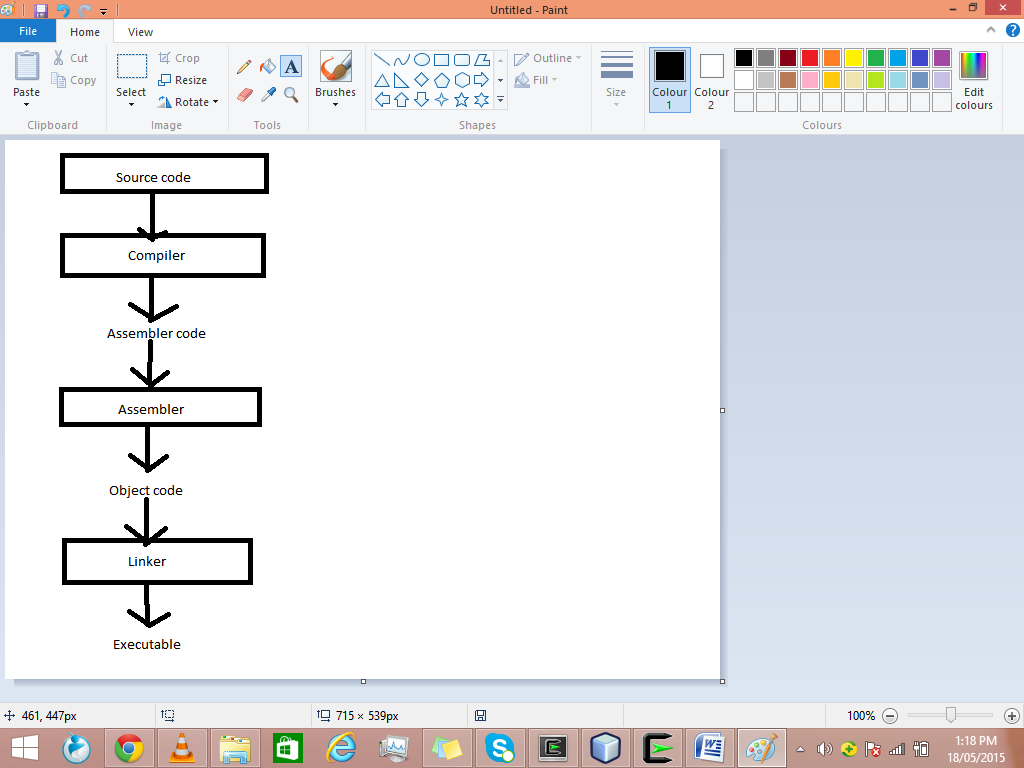
Build tools help building a program. They have to manage various things like:

-version control

-code quality

-compilation and linking

Tool Chain



Compiler- converts a source code(code written in some programming language) to an assembly language code. It performs

-lexical analysis

-parsing

-semantic analyses

-code optimisation

Assembler- converts assembly language code to an object code.

Linker- takes one or more object files created by assembler and convert it into a single executable file. There are two types of linkers

* Dynamic linker-it copies the undefined symbols along with the libraries where they are defined.
* Static linker- it copies the library routes in the executable image.