Dipen Parmar

FlexSass Website Template

Department of computer SCIENCE, Atmiya University

master of science in information techology

2019

**HARDWARE REQUIREMENTS**

**Technical Details Hardware (**Requirements**)**

|  |  |
| --- | --- |
| **Particulars** | **Hardware**  **Requirements** |
| Processor Brand | Intel, Amd etc. |
| Processor Type | I3 Processor |
| Processor Speed | 1.5 GHz |
| RAM Size | 2 GB |
| Memory Type | DDR3 |
| Hard Drive Size | 500 GB |

**Technical Details Software (**Requirements**)**

|  |  |
| --- | --- |
| **Particulars** | **Software Requirements.** |
| Operating System | Windows, Linux, Mac, any |
| Browser | • Chrome\* 36+ • Edge\* 20+ • Mozilla Firefox 31+ |

WireFrame Link <https://s3.amazonaws.com/assets.mockflow.com/app/wireframepro/fileexport/Export_D7b7b0c913e15f3a4e1cddefa1287e431.pdf>

Getting Started

* Project Structure

├───dest

│ ├───assets

│ │ ├───img

│ │ │ └───SVG

│ │ ├───script

│ │ └───style

│ │ ├───artical

│ │ ├───dashboard

│ │ ├───feature\_section

│ │ ├───landing\_page

│ │ ├───modern\_site

│ │ ├───nest

│ │ └───nest2

│ ├───img

│ └───sass

│ └───landing\_page

└───src

├───img

│ └───SVG

├───sass

│ ├───artical

│ ├───dashboard

│ ├───feature\_section

│ ├───landing\_page

│ ├───modern\_site

│ ├───nest

│ │ ├───abstracts

│ │ ├───base

│ │ ├───components

│ │ ├───layout

│ │ └───pages

│ └───nest2

└───script

Css FlexBox

* FlexBox Basics & Terminology

The Flexible Box Module, usually referred to as flexbox, was designed as a one-dimensional layout model, and as a method that could offer space distribution between items in an interface and powerful alignment capabilities. This article gives an outline of the main features of flexbox, which we will be exploring in more detail in the rest of these guides.

When we describe flexbox as being one dimensional we are describing the fact that flexbox deals with layout in one dimension at a time — either as a row or as a column. This can be contrasted with the two-dimensional model of [*CSS Grid Layout*](https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Grid_Layout), which controls columns and rows together.

* The two axes of flexbox

When working with flexbox you need to think in terms of two axes — the main axis and the cross axis. The main axis is defined by the [flex-direction](https://developer.mozilla.org/en-US/docs/Web/CSS/flex-direction) property, and the cross axis runs perpendicular to it. Everything we do with flexbox refers back to these axes, so it is worth understanding how they work from the outset

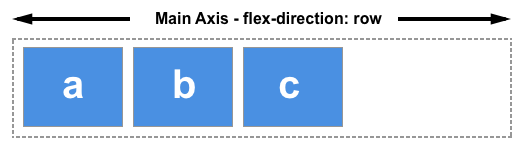
* [The Main axes of flexbox](https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Flexible_Box_Layout/Basic_Concepts_of_Flexbox#The_two_axes_of_flexbox)

The main axis is defined by flex-direction, which has four possible values:

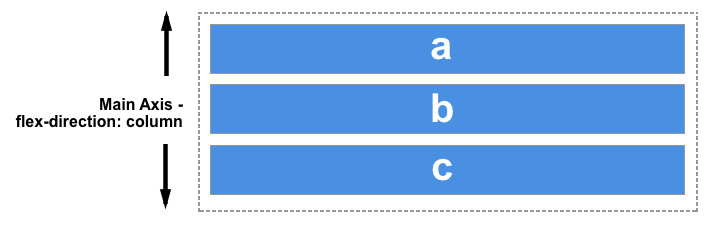
* row
* row-reverse
* column
* column-reverse

Should you choose row or row-reverse, your main axis will run along the row in the **inline direction**

.



Choose column or column-reverse and your main axis will run from the top of the page to the bottom — in the **block direction**

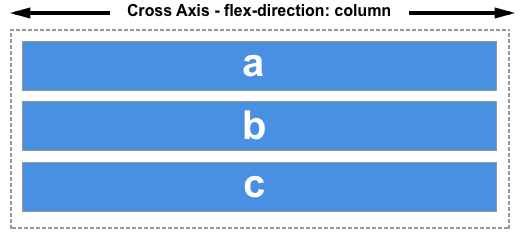


* The cross axis of flexbox

The cross axis runs perpendicular to the main axis, therefore if your flex-direction (main axis) is set to row or row-reverse the cross axis runs down the columns.

****

If your main axis is column or column-reverse then the cross axis runs along the rows.



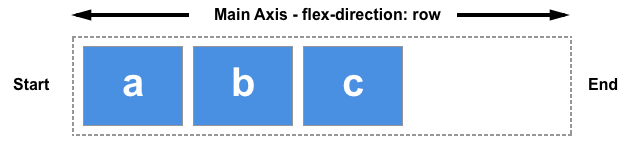
Understanding which axis is which is important when we start to look at aligning and justifying flex items; flexbox features properties that align and justify content along one axis or the other.

* Start and end lines (MainAxis)

Another vital area of understanding is how flexbox makes no assumption about the writing mode of the document. In the past, CSS was heavily weighted towards horizontal and left-to-right writing modes. Modern layout methods encompass the range of writing modes and so we no longer assume that a line of text will start at the top left of a document and run towards the right hand side, with new lines appearing one under the other.

You can read more about the relationship between flexbox and the Writing Modes specification in a later article, however the following description should help explain why we do not talk about left and right and top and bottom when we describe the direction that our flex items flow in.

If the flex-direction is row and I am working in English, then the start edge of the main axis will be on the left, the end edge on the right.



If I were to work in Arabic, then the start edge of my main axis would be on the right and the end edge on the left.



In both cases the start edge of the cross axis is at the top of the flex container and the end edge at the bottom, as both languages have a horizontal writing mode.

After a while, thinking about start and end rather than left and right becomes natural, and will be useful to you when dealing with other layout methods such as CSS Grid Layout which follow the same patterns.

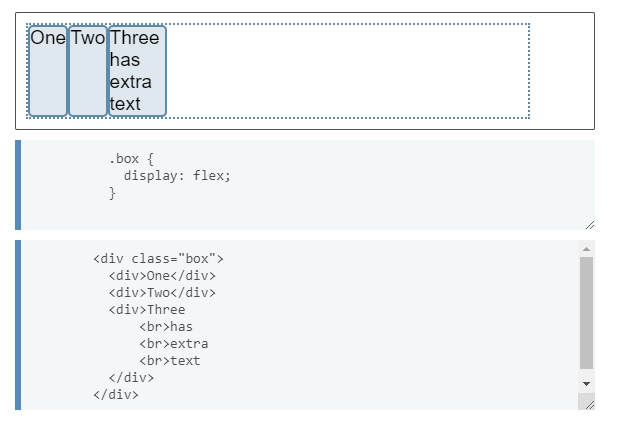
* The flex Container

An area of a document laid out using flexbox is called a **flex container**. To create a flex container, we set the value of the area's container's [display](https://developer.mozilla.org/en-US/docs/Web/CSS/display) property to flex or inline-flex. As soon as we do this the direct children of that container become **flex items**. As with all properties in CSS, some initial values are defined, so when creating a flex container all of the contained flex items will behave in the following way.

* Items display in a row (the flex-direction property's default is row).
* The items start from the start edge of the main axis.
* The items do not stretch on the main dimension, but can shrink.
* The items will stretch to fill the size of the cross axis.
* The [flex-basis](https://developer.mozilla.org/en-US/docs/Web/CSS/flex-basis) property is set to auto.
* The [flex-wrap](https://developer.mozilla.org/en-US/docs/Web/CSS/flex-wrap) property is set to nowrap.

The result of this is that your items will all line up in a row, using the size of the content as their size in the main axis. If there are more items than can fit in the container, they will not wrap but will instead overflow. If some items are taller than others, all items will stretch along the cross axis to fill its full size.

You can see in the live example below how this looks. Try editing the items or adding additional items in order to test the initial behavior of flexbox.

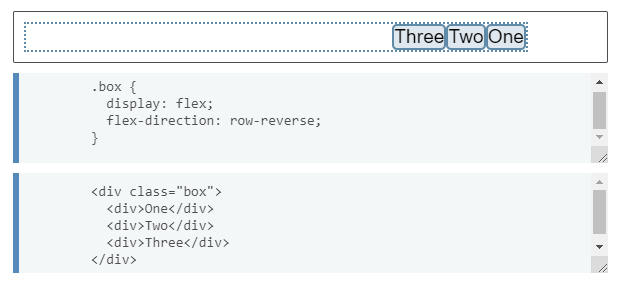


* Changing flex-direction

Adding the [flex-direction](https://developer.mozilla.org/en-US/docs/Web/CSS/flex-direction) property to the flex container allows us to change the direction in which our flex items display. Setting flex-direction: row-reverse will keep the items displaying along the row, however the start and end lines are switched.

If we change flex-direction to column the main axis switches and our items now display in a column. Set column-reverse and the start and end lines are again switched.

The live example below has flex-direction set to row-reverse. Try the other values — row, column and column-reverse — to see what happens to the content.

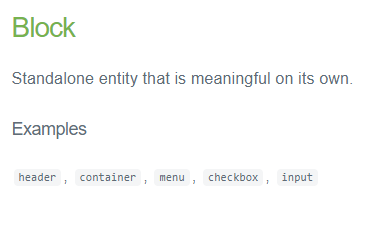


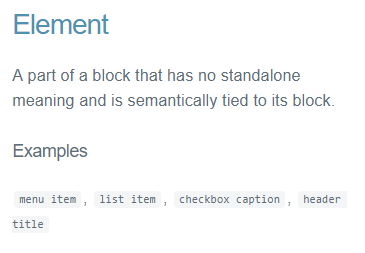
BEM

* What is BEM (Blocks, Elements and Modifiers)

CSS is a language that is easy to learn but very hard to maintain. As the project grows larger, without proper structure, maintaining CSS is unbearable. People have come up with different types of solutions to this such as OOCSS, SMACSS, and BEM. Currently, BEM is the most widely used, it’s unique naming method makes CSS to maintain. Without further a due, let’s start learning BEM.

BEM is a naming styling that is created by Yandex (think of them as Russia’s Google). The problem BEM is trying to solve is the naming problem and structure that CSS often run into. BEM also provides a better structure for your CSS code and scalable CSS.



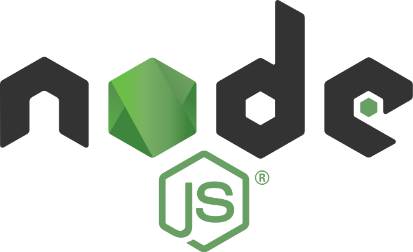




* Benefits of BEM Methodology
  + **Modularity**
    - Block styles are never dependent on other elements on a page, so you will never experience problems from cascading.
    - You also get the ability to transfer blocks from your finished projects to new ones.
  + **Reusability**
    - Composing independent blocks in different ways, and reusing them intelligently, reduces the amount of CSS code that you will have to maintain.
    - With a set of style guidelines in place, you can build a library of blocks, making your CSS super effective.
  + **Structure**
    - BEM methodology gives your CSS code a solid structure that remains simple and easy to understand.

**Getting Started**

You will need a working version of Node and NPM to get started.

****

NodeJs

* What is Node.js?
  + Node.js is an open source server environment
  + Node.js is free
  + Node.js runs on various platforms (Windows, Linux, Mac OS X, etc.)
  + Node.js uses JavaScript on the server
* Why Node.js?

## Node.js uses asynchronous programming!

Here is how PHP or ASP handles a file request:

* + Sends the task to the computer's file system.
  + Waits while the file system opens and reads the file.
  + Returns the content to the client.
  + Ready to handle the next request.

Here is how Node.js handles a file request:

* + Sends the task to the computer's file system.
  + Ready to handle the next request.
  + When the file system has opened and read the file, the server returns the content to the client.
* What Can Node.js Do?
  + Node.js can generate dynamic page content
  + Node.js can create, read, write, delete, and close files on the server
  + Node.js can collect form data
  + Node.js can add, delete, modify data in your database
* What is a Node.js File?
  + Node.js files contain tasks that will be executed on certain events
  + A typical event is someone trying to access a port on the server
  + Node.js files must be initiated on the server before having any effect
  + Node.js files have extension ".js"

****

Node Package Manager

## What is NPM?

* + Node.js is an open source server environment
  + Node.js is free
  + Node.js runs on various platforms (Windows, Linux, Mac OS X, etc.)
  + Node.js uses JavaScript on the server

## Why Node.js?

* + NPM is a package manager for Node.js packages, or modules if you like.
  + [www.npmjs.com](https://www.npmjs.com/) hosts thousands of free packages to download and use.
  + The NPM program is installed on your computer when you install Node.js

## What is a Package?

* + A package in Node.js contains all the files you need for a module.
  + Modules are JavaScript libraries you can include in your project.

## Use npm to. .

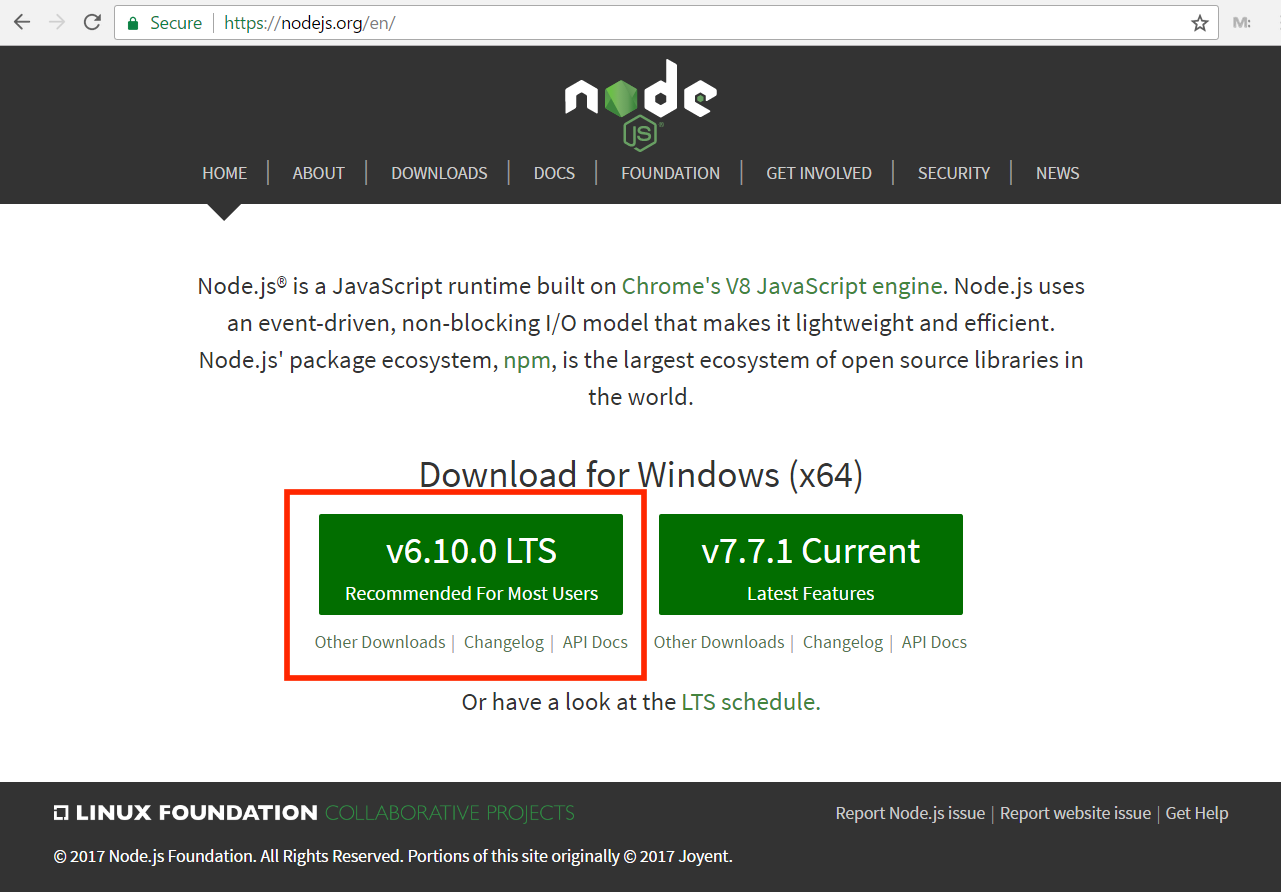
* + Adapt packages of code for your apps, or incorporate packages as they are.
  + Download standalone tools you can use right away.
  + Run packages without downloading using [npx](https://www.npmjs.com/package/npx).
  + Share code with any npm user, anywhere.
  + Restrict code to specific developers.
  + Create Orgs (organizations) to coordinate package maintenance, coding, and developers.
  + Form virtual teams by using Orgs.
  + Manage multiple versions of code and code dependencies.
  + Update applications easily when underlying code is updated.
  + Discover multiple ways to solve the same puzzle.
  + Find other developers who are working on similar problems and projects.

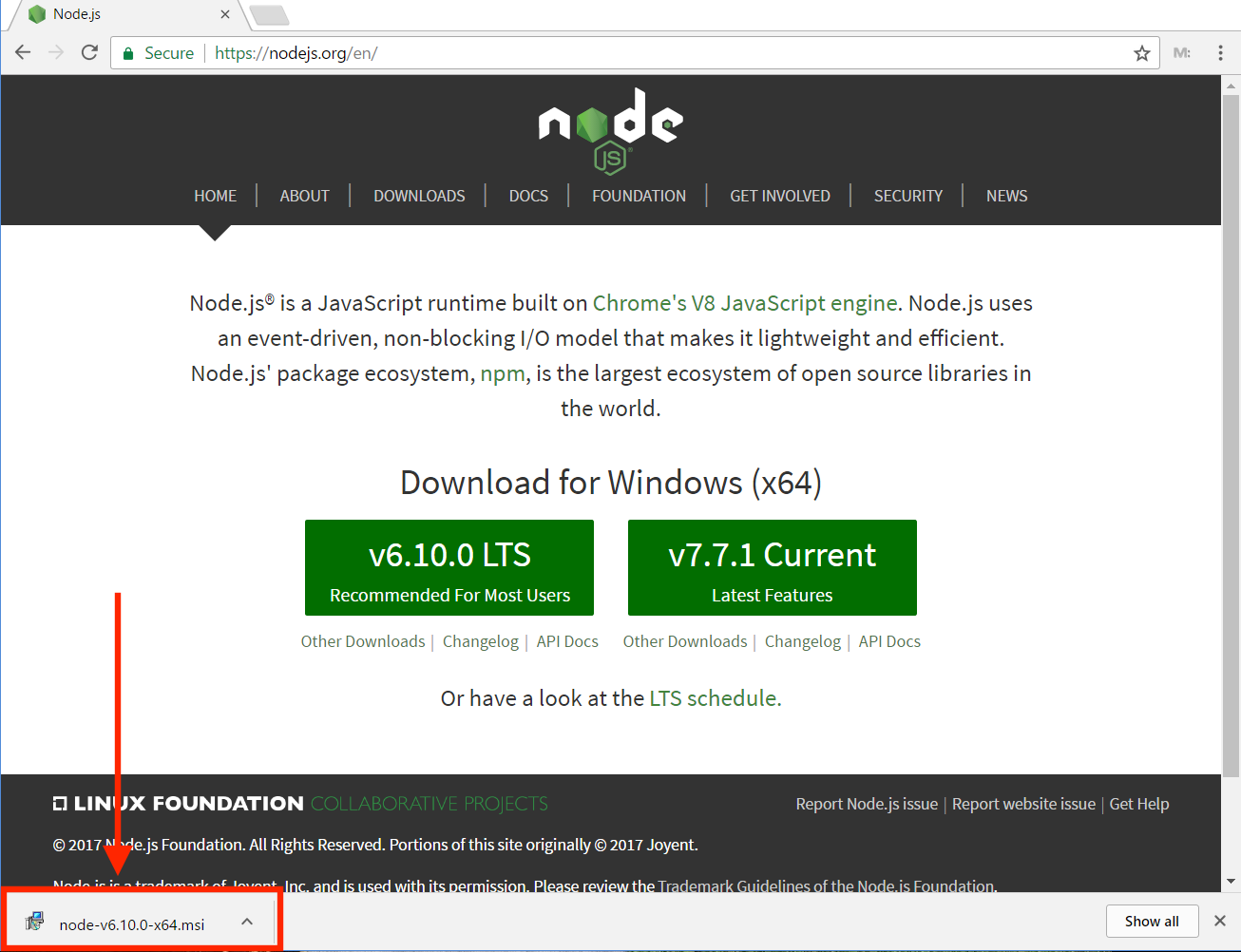
## Sharing packages and collaborating with others

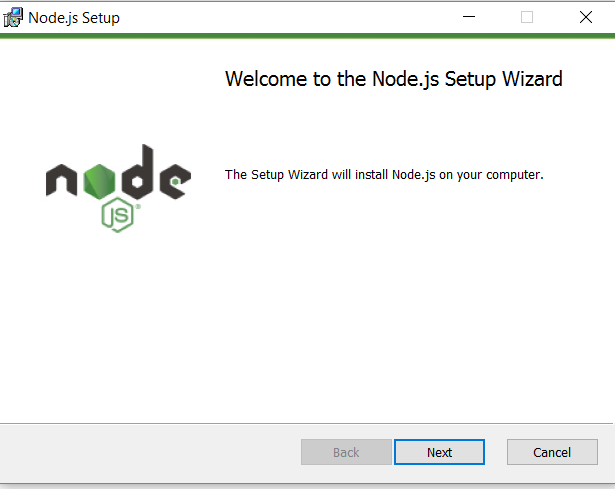
If you choose to share your packages publicly, there is no cost. To use and share private packages, you need to upgrade your account. To share with others, create organizations, called [**npm Orgs**](https://docs.npmjs.com/orgs), and invite others to work with you, privately (for a fee) or publicly (for free). Or you can sign up for a private instance of npm for your company, called [**npm Enterprise**](https://npmjs.com/enterprise), so you can develop packages internally that are not shared publicly.

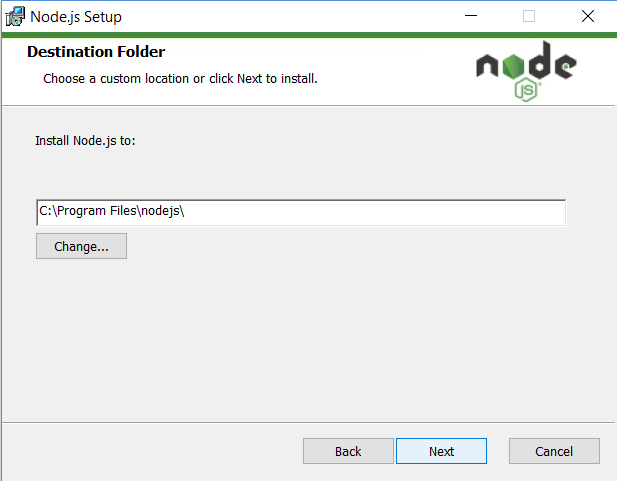
**Install Node.js & NPM on Windows**

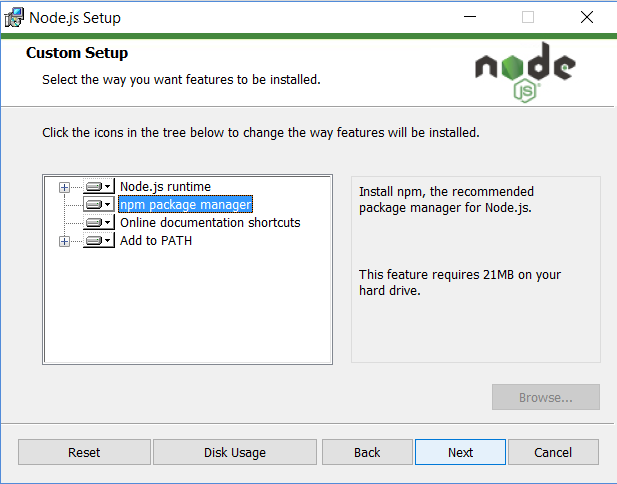
Go to the [official Node.js site](https://nodejs.org/) which will offer you two versions of Node to download. Click on the Installer that says “Recommended for Most Users”.

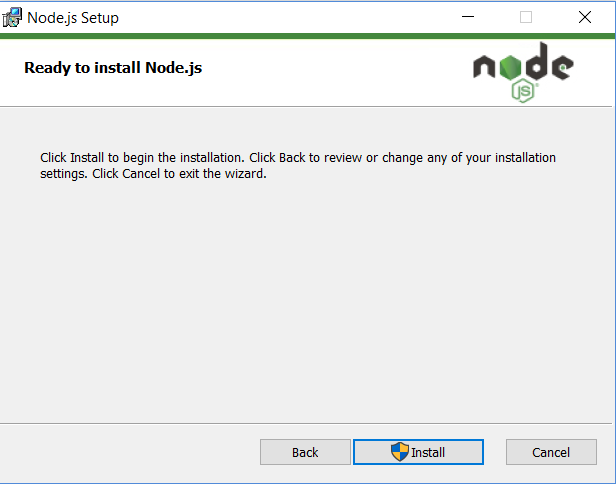




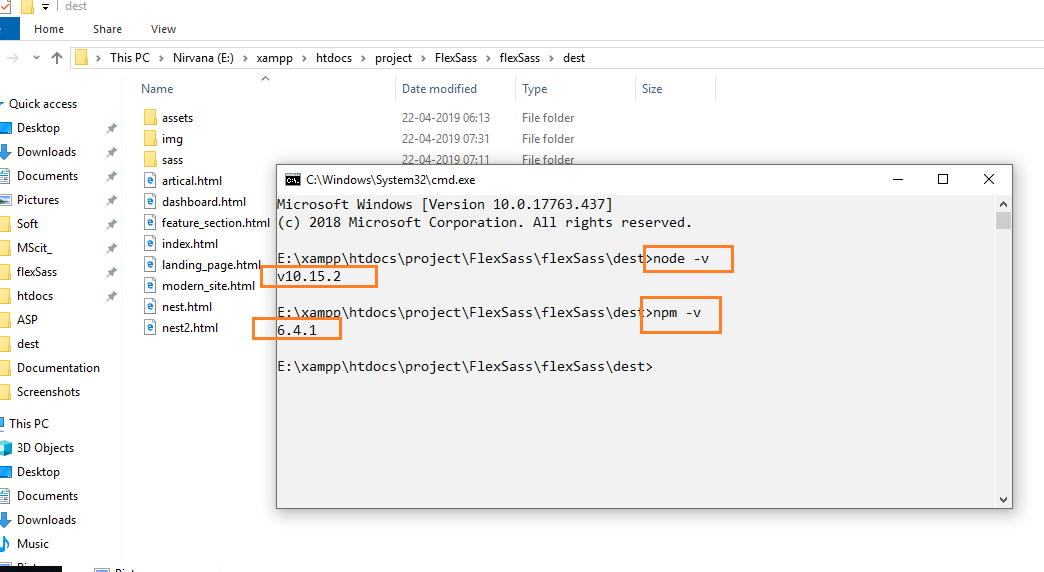








In The Command Prompt. To confirm Node is installed type node –v which will print the current version number. To confirm NPM is installed type npm -v which will print its current version number.





Gulp

* What is Gulp

Gulp is a JavaScript task runner that lets you automate tasks such as…

* + Bundling and minifying libraries and stylesheets.
  + Refreshing your browser when you save a file.
  + Quickly running unit tests
  + Running code analysis
  + Less/Sass to CSS compilation
  + Copying modified files to an output directory

------------------------------------------------------------------------------------

* + Stream-based build system
  + NodeJs
  + Code (JavaScript) over configuration
  + Asynchronous
  + Simple API
  + Plugin based
  + Easy
* The simple gulp API

Using gulp is super simple because you don’t have to figure out how a complex API works in order to be productive with it. There are only 4 API’s in gulp!

| API | Purpose |
| --- | --- |
| gulp.task | Define a task |
| gulp.src | Read files in |
| gulp.dest | Write files out |
| gulp.watch | Watch files for changes |

* Installing gulp via npm

The npm package manager comes installed with Node.js. While node.js isn’t a requirement to use gulp, it does make demonstrating it a lot easier. I will be installing gulp from npm locally into my project. Make sure that you’re in your project’s root folder before running the command, otherwise your node modules will be downloaded into the wrong folder

* + **cd myproject**
  + **npm install --save-dev gulp**

This will install the gulp node module locally to the project (as opposed to globally). The --save-dev argument lets npm know to update it’s package.json file with a new devDependencies record. devDependencies will need to be resolved at development time, where as dependencies will need to be resolved at run time. Because gulp is a tool to aid us in development, it needs to be resolved at development time.

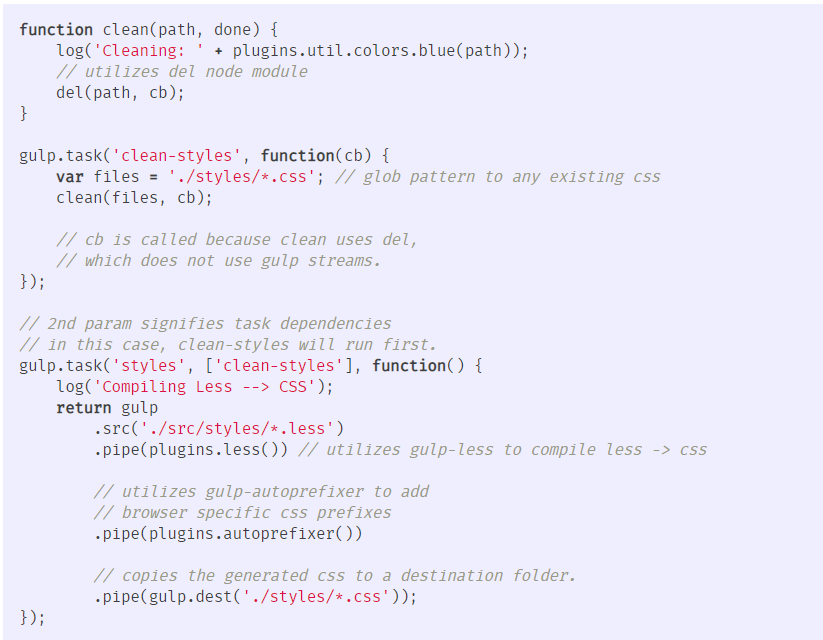
* Creating a gulpfile

A gulpfile is a file that will act as a manifest to define our tasks. Tasks that we want to execute will be found within this file. Whenever we run the command **gulp hello-world** from the command line, we are telling gulp that we want to run the hello-world task within gulpfile.js. After creating **gulpfile.js** within the root of your project, add a basic tasks.  
 

**require** is a function implemented by node (which is an implementation of the [CommonJS spec](http://wiki.commonjs.org/wiki/Modules/1.1.1)) that will add references to node modules that we have installed. Once we make a reference to the gulp module, we can use it to create a task. Here, our task simply writes to the console window, but you could have it do any number of automated tasks.

* A practical example

Create a gulp task (or set of gulp tasks) for cleaning directories and compiling less to CSS. Such a set of tasks may look like the following. This set of tasks also has its own set of dependencies, which would require using npm install to get installed locally



Running the following in the command line starts the watch:

**$ gulp styles**

Hopefully it’s obvious why taking the time to create gulp tasks for automating repetitive development tasks can increase productivity.