Course: WebStorm Fundamentals

===

Ctrl + Shift + Enter completes a statement. Use this religiously. Ex:

---

var animal = function <Ctrl + Shift + Enter>

Result:

var animal = function () {

};

After you type in the parameters to the function, do ctrl+shift+enter again to place the cursor inside the function body.

---

console.log <Ctrl + Shift + Enter>

Result:

console.log(|); (the cursor is inside the log function)

---

var x = "John|" <Ctrl + Shift + Enter>

Result:

var x = "John";

---

console.log("Bill|"); <Ctrl + Shift + Enter>

Result:

Cursor jumps to the next line

===

Double shift (shift + shift) gives you a search-all (including files/folders)

Ctrl + Shift + N gives you the search menu for your project directory.

On your keyboard, hit the End key to go to the end of the current line. The Home button goes to the beginning of the current line.

===

Create custom scopes -> This will make it so much easier to navigate between files!

Can match to a pattern such as file\*.css (matches any file with the css extension)

file\*.css | file\*.less

===

Live editing only works with the Chrome browser. Also need the Jetbrains IDE support extension.

Check the option "Restart if hotswap fails"

Need to edit configurations (there's an icon for it) in order to set up a debugging session

When doing a live edit, put webstorm and your browser side by side. When you put the mouse on an html tag, it will highlight the associated block on the webpage.

Live editing CSS doesn't work quite as well, good idea to go in settings and select the auto-update option

===

Can drag the cursor to multiple lines using alt + drag

Can also hold alt and use mouseclicks to have multiple cursors in exactly the same way as Sublime.

Ctrl + period allows you to collapse the block of code surrounding your cursor (ex/ collapse a div)

To surround your selected code with a tag: ctrl + alt + t, then select emmet from the menu that appears.

If you double click a tag name, renaming it will change both the opening and closing tag for you.

Alt + J does what Ctrl + D does in Sublime

You can find boilerplate shortcuts (custom emmet style commands) under Settings -> Editor -> Live Templates

===

link:css tab creates <link rel="stylesheet" href="style.css"/>

When filling in the href, try ctrl + spacebar \*2 to get better recommendations.

Can also type the name of the file and then ctrl + spacebar\*2 for it to auto-fill the path (if there's only 1 suggestion)

Yet another way to do this is to right click on the desired file, click copy reference, then paste it in

===

If you create a new file using an already provided extension, the new file will automatically have some boilerplate code.

Ex/ New -> HTML file -> new file has HTML boilerplate

===

ctrl + alt + L reformats the code. Similar to line -> re-indent in Sublime

If you hover your mouse in the top-right corner, some browser icons will appear. Your webpage will actually be hosted on localhost!

CSS colors are previewed to the left

You can manually trigger intelliSense with ctrl + space

Webstorm seems to enforce indentation (or maybe it was ctrl + alt + l that made that happen?)

If you click on a breadcrumb, you can see the different files and folders belonging to that breadcrumb

In the bottom-right corner, you can see which line AND column you are on. If you click on it, you can even change what line and column the cursor is at.

Can also set the document to read-only. Can even vary the highlighting level (the amount of syntax checking don for you)

In the project side tab, there's a little gear icon that lets you change display settings for that tab. A cool one is unchecking pinned mode. When you do this, the project menu hides when you click away from it.

You can drag the tabs to different places, even the top of the screen.

When inside any menu, hitting escape will always return you to the editor

If you click the structures tab inside an html document, you can navigate the tree structure of the document!

The event log is what logs any IDE specific messages such as errors and problems with plugins.

If you write a TODO comment in your code, it gets logged in the TODO tab!

In webstorm's terminal, you can click the plus symbol on the top-left to make multiple terminal instances!

===

To change tab size (or other settings) in Javascript, go to Settings -> Editor -> Code Style -> JavaScript

Change fonts in Settings -> Editor -> Colors & Fonts -> Font

===

Help -> Find Action (or ctrl + shift + a). Very useful for finding out things like how to zoom, and also to change settings.

===

When making new key bindings, the IDE automatically tells you if there will be conflicts, and what the conflicting key binding is.

Can assign multiple key bindings to the same action.

When you change a key binding, it gets saved to a custom profile. This is to preserve the default (and other) keymap profiles.

===

When using Emmet (ex/ ul>li\*5 tab), the cursor will blink red in the places it expects content. You can hit enter or tab to navigate through these.

===

In the project menu, alt + insert lets you add a new file (instead of right clicking)

===

Unlike Sublime, you can simply drag files in the project menu to move them. What's even cooler is that it will automatically change references to that file (ex/ script src="path")!

===

Typing in CSS is extremely easy because IntelliSense actually auto-inserts semi-colons and puts your cursor after it (and other useful guiding actions)

If you click the little color square in the CSS, you actually get a color wheel!

If you type in "ib", intellisense suggests inline-block

===

You should use watchers. In the case of .less files, it automatically generates a .css file from it.

npm install -g less

gem install sass

===

us + tab -> use strict;

===

\*\*\*\*\*\*\*\*\*\*

javascript

\*\*\*\*\*\*\*\*\*\*

Right click, refactor is amazing

You can document your functions by typing /\*\* + enter. This will autofill @param, and whatever you have documented here will appear in IntelliSense (or at least when you hover over the function).

Can even type @param {Boolean} validate to further improve the documentation.

If you check the lightbulb icon next to a warning, you can do things like "update JSDoc" and "convert parameters to object" (really useful)

===

When you see the lightbulb icon, hit Alt + Enter to upon up its menu (instead of having to click on it)

===

If you put the cursor on a function and hit Ctrl + B, you can see where the function was originally defined (even if it's in a different file). Can alternatively use Ctrl + Click.

Also "Find Usages" if you right click

===

Language injection is awesome. You can be in a javascript file with a variable set to a string of html code. Use the lightbulb suggestion and you can format the html in a separate window.

Lightbulb -> Alt + Enter, Inject Language or Reference

Do it again and select the option to Edit Code Fragment

A cool application of this is to inject a regular expression. You are then given an option to check it with a simple test window.

===

Can put conditions on your debugging breakpoints (only break if...)