ASSIGNMENT COVERSHEET

Prague College

Student Name: Arash Banihashemi	
Class: Website Design	
Assignment: A1 – "Walk&Learn" Website Development	
Lecturer: David Petryca	Semester: 1404
Due Date: 19/11/2014	Actual Submission Date: 19/11/2014

Evidence Produced (List separate items)	**Location (Choose one)**	
PDF	X	Uploaded to the Learning Center (Moodle)
		Submitted to reception
Note: Email submissions to the lecturer are not valid.		

Student Declaration	
I declare that the work contained in this assignment was researched and prepared by me, except where acknowledgement of sources is made. **I understand that the college can and will test any work submitted by me for plagiarism.** **Note:** The attachment of this statement on any electronically submitted assignments will be deemed to have the same authority as a signed statement	
Date: 19/11/2014	Student Signature: Arash Banihashemi

A separate feedback sheet will be returned to you after your work has been graded.
Refer to your Student Manual for the Appeals Procedure if you have concerns about the grading decision.

Student Comment (Optional)
Was the task clear? If not, how could it be improved?
Was there sufficient time to complete the task? If not, how much time should be allowed?
Did you need additional assistance with the assignment?
Was the lecturer able to help you?
Were there sufficient resources available?
How could the assignment be improved?
For further comments, please use the reverse of this page.

CSS FLEX SYSTEM RESEARCH

Arash Banihashemi

WEBSITE DESIGN

Contents

Introduction

Historically the process of creating multi-column layout with HTML and CSS has always come with a few headache when your layout your columns using float for example you have to deal with issue like collapsing background and mismatched columns height. There is a way to practically eliminate all these headache using only html and CSS. CSS flex box layout model which allows us to layout our content with more flexibility than we've ever seen before in HTML and CSS.

In this assignment I'm going to explain how to use the flex box model to easily switch back and forth between horizontal and vertical layout and also explain how to adjust alignment of our contents including the ability to vertically center our content using only a line or two of CSS code step by step.

Flexbox Layout Container:

As you can see in below HTML, we have a Container DIV and four separated DIV.

`<div class="container">` : "container" is our container class Name,
`<div class="item item1">ITEM 1</div>` "item" is a name of class for our multiple items inside the container and Item1 is name of our second class

By default DIV is a block level element which means there are arranged vertically and also inline elements on the other hands is arranged horizontally or side by side but in flex elements which can be arranged vertically or horizontally

HTML:

```
<!DOCTYPE html>
<html>
<head>
	<title>CSS Flexbox</title>
	<link href="styles.css" rel="stylesheet" />
</head>
<body>
	<div class="container">
		<div class="item item1">ITEM 1</div>
		<div class="item item2">ITEM 2</div>
		<div class="item item3">ITEM 3</div>
		<div class="item item4">ITEM 4</div>
	</div>
</body>
</html>
```

CSS:

.container: container is a parent class of all items and actually items inside the container are immediate children of that container which all div that we've put inside it.

.item: items don't have class of inline or block or inline-block or anything like that

```
.container {
      background-color: #555;
      width: 800px;
      height: 400px;
      margin: 0 auto;
      display: flex;
}

.item {

}

.item1 { background-color: red; }
.item2 { background-color: blue; }
.item3 { background-color: green; }
.item4 { background-color: orange; }
```

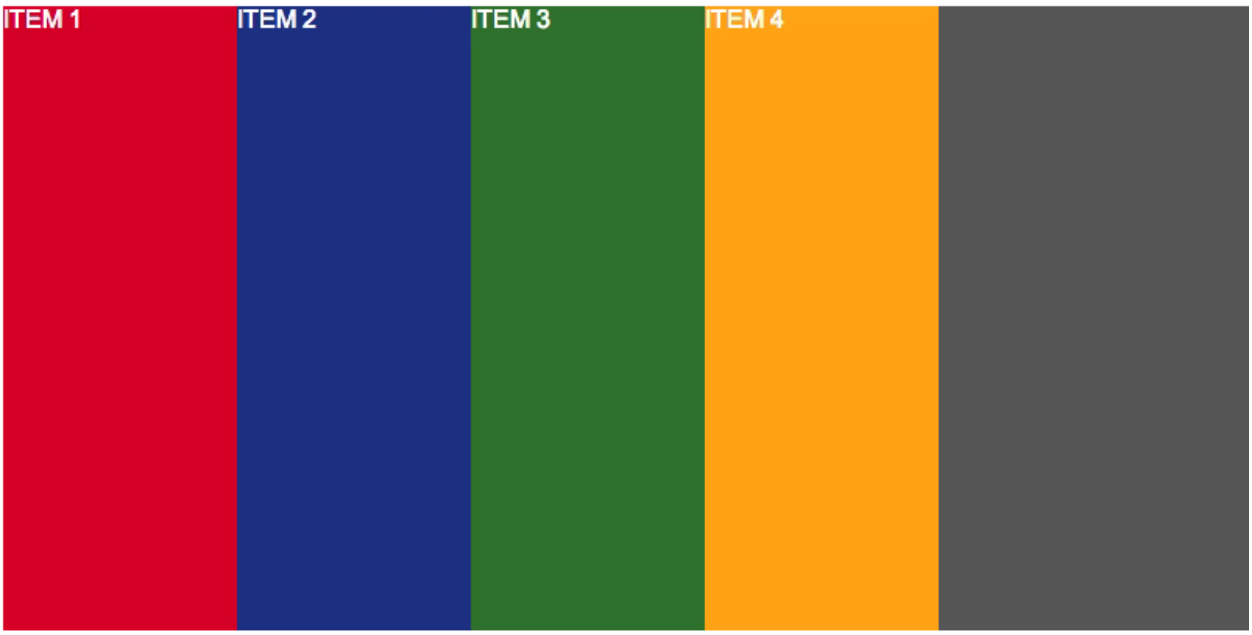
Column Widths:

In Flexbox each of items only takes up as much width as content that inside it and by default it takes up full height but width is limited to the content that inside that div. One of the best thing about flexbox layout model is the flexibility that gives us to size and arrange our content.

The CSS **flex-basis** property specifies the flex basis which is the initial main size of a flex item. The property determines the size of the content-box unless specified otherwise using box-sizing.

```
flex-basis: 400px;
```

I put above code in .item and our box got change to below picture. In fact flex-basis is natural width of these columns before any of the flex property is start to kick in.



The CSS **flex-grow** property specifies the flex grow factor of a flex item.

```
flex-grow: 1;
```

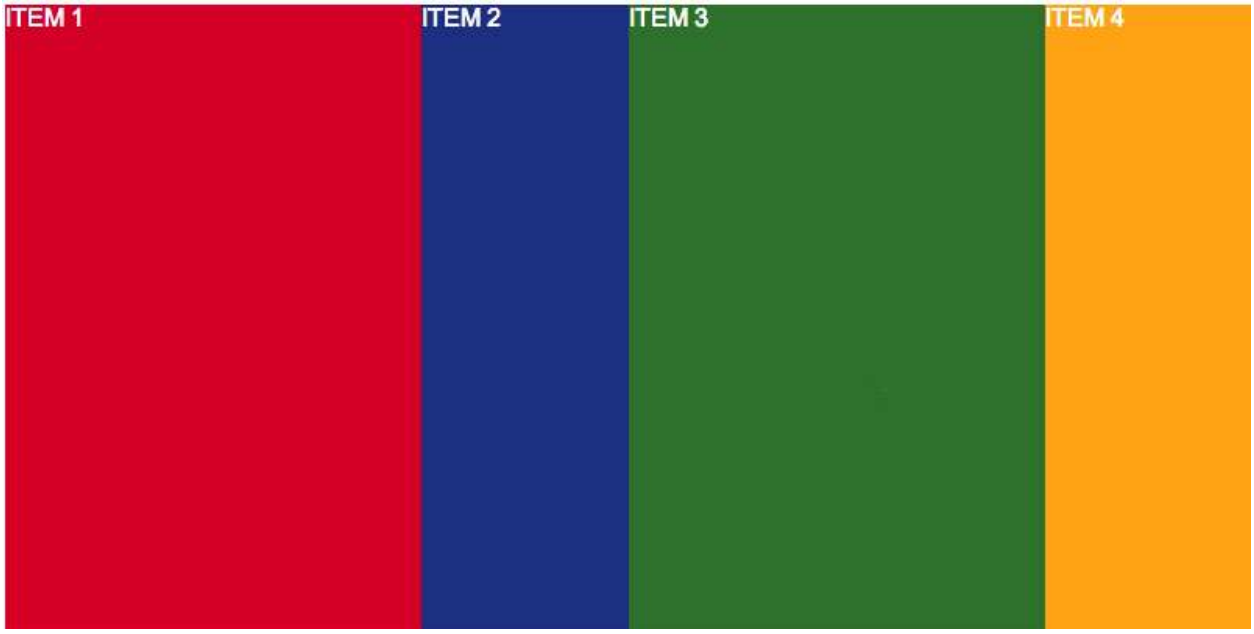
if I put flex-grow: 1 inside .item , it will be effect on each item and actually every item gets same value. The value of flex-grow is completely arbitrary and if I want to do give all four of these columns "100" and it will give us exact same result. Because what is doing is comparing this value to the same flex-grow value in the sibling elements. Flex-grow determine how much that particular item and particular items is going to grow in relation to the sibling column.

Flex Direction:

Flex Direction allows us to determine whether we want to create columns or rows of information. Flex Direction property need to be applied to the .container and what the direction property does, tells that container how to layout the items within the container.

```
flex-direction: row;
```

In above code we can set direction to "row" or "column". If we set as row, we will have bellow thing.




```
flex-direction: column;
```

But if we set to column, we will have below box in other direction

Align-Items

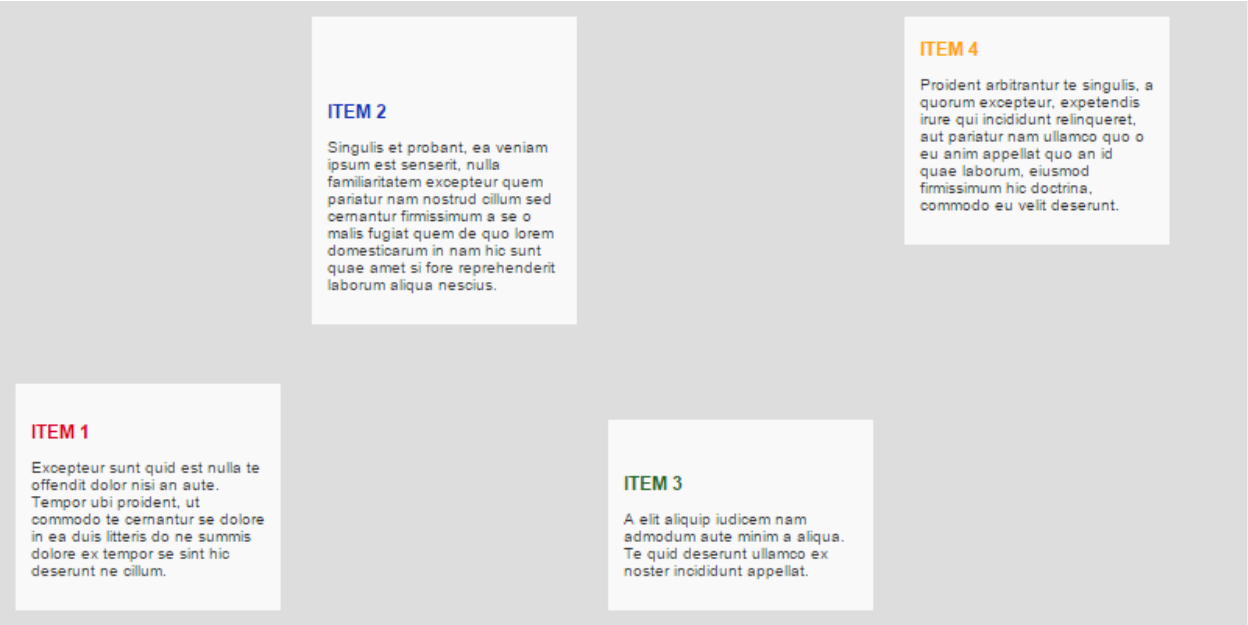
Another way we can align or arrange our content within flex box container. It’s a way to align items vertically in the container based on height those items. The default value for those item in our browser is “stretch”.

`align-self:`

It makes possible to override the align-items value for specific flex items.

flex-start: cross-start margin edge of the item is placed on the cross-start line

flex-end: cross-end margin edge of the item is placed on the cross-end line



Browser Compatibility

IE	Firefox	Chrome	Safari	Opera
8	30	29		
9	31	36		12.1
10	32	37	7	24
11	33	38	8	25
	34	39		26
	35	40		27
	36	41		