

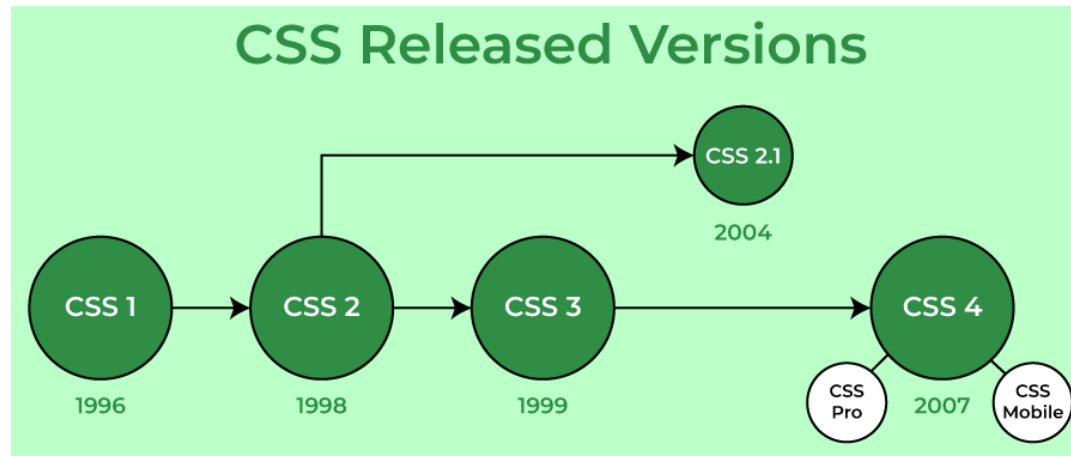
CSS

# 내 용

- CSS 개요
- CSS 정의 방법
- CSS Syntax
- CSS 선택자 (Selector)
- Box Model
- Layout (flex, grid, table, z-index)
- Block, Inline, inline-block
- Display, visibility, opacity
- CSS Transition, Translation, Animation
- Viewport, Media query

# CSS 개요

- Cascading Style Sheet
- 문서를 표시하는 방법을 지정하는 언어 : 스타일, 레이아웃
- 규칙 기반 언어 : 웹 페이지의 특정 요소 또는 요소 그룹에 적용할 스타일 그룹을 지정하는 규칙을 정의



# 웹 페이지 구성 요소

- 콘텐츠 : 정보
- Tag
  - 콘텐츠 구조, 내용
- CSS
  - 스타일 지정
  - 애니메이션
- JavaScript
  - 동적 : 이벤트, 애니메이션
  - 인터랙티브(상호) 작용 : 이용자, 서버

# JavaScript와 CSS 역할 분담






- 애니메이션등의 동적 처리를 브라우저에서 지원하게 됨
  - 전에는 자바스크립트에서 전적으로 처리하여 부담
  - 브라우저의 CSS3의 그래픽 처리 기능이 보다 효율적
    - 하드웨어 가속으로 작동
- 구형 브라우저의 CSS 미 지원 기능은 자바스크립트로 처리
  - jQuery 등의 자바스크립트 라이브러리를 활용하면 더 쉽게 가능

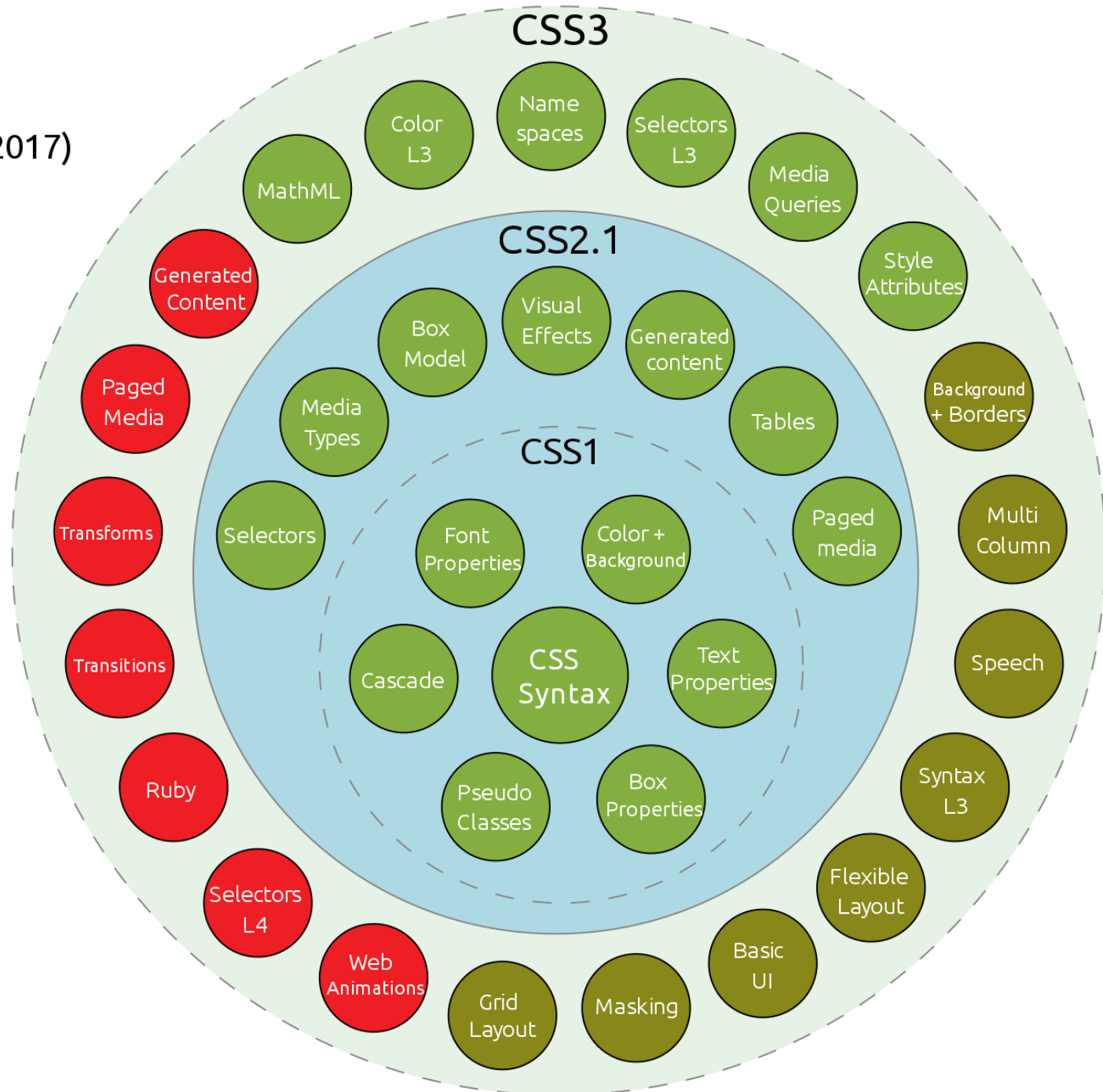
# CSS 기능

- 웹 페이지의 레이아웃(layout)과 포맷 설정
- 엘리먼트의 포맷 설정
  - Color, Font, Size of text
- 엘리먼트간의 간격 조정
  - Box model
- 엘리먼트의 위치와 배치
- Background images/colors
- 뷰포트(view port )와 미디어 쿼리
  - Different displays for different devices and screen sizes
- 애니메이션

# CSS3

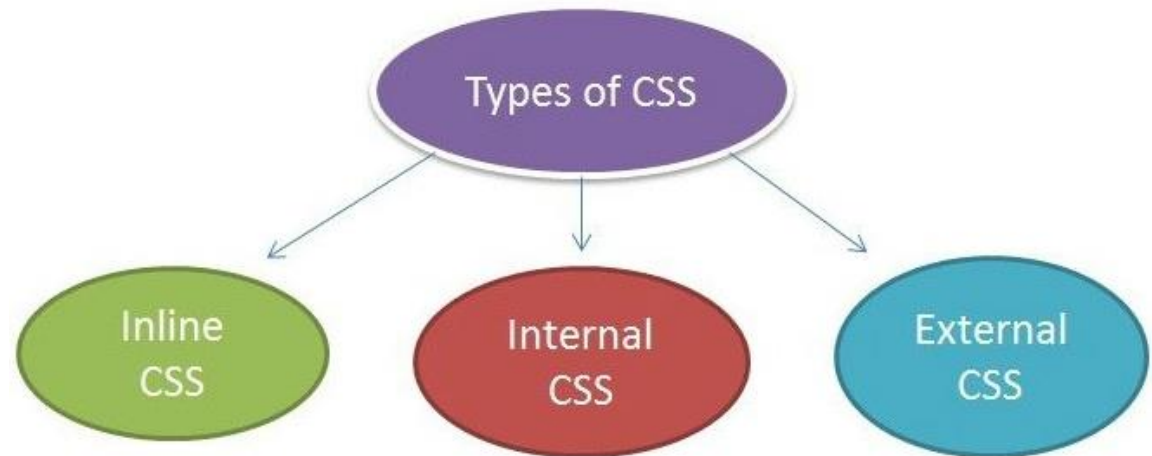
Taxonomy & Status (September 2017)

-  W3C Recommendation
-  Candidate Recommendation
-  Last Call
-  Working Draft
-  Obsolete or inactive



# CSS (Cascading Style Sheet) 위치

- CSS 위치
  - 태그 내부에 선언 (inline)
  - 웹 문서의 head에 선언 (internal)
  - 웹 문서와 분리된 외부 CSS 파일 (external)





# CSS (Cascading Style Sheet) 적용

- Cascading
  - 부모의 스타일을 자식이 물려받아 적용
  - 순차적으로 적용하여 최종의 것으로 적용
- Style 적용 순서
  - 엘리먼트의 default 스타일 적용
  - OS Style 적용
  - Browser Style 적용
  - 부모의 스타일 적용
  - 헤드에 정의된 스타일 적용 (불러들인 외부 스타일 파일 포함)
  - 태그에 정의된 스타일 적용

```
<div> 부모 엘리먼트  
  <p>Hi</p> 자식 엘리먼트  
  <p>Hi</p> 자식 엘리먼트  
</div>
```

# Inline CSS

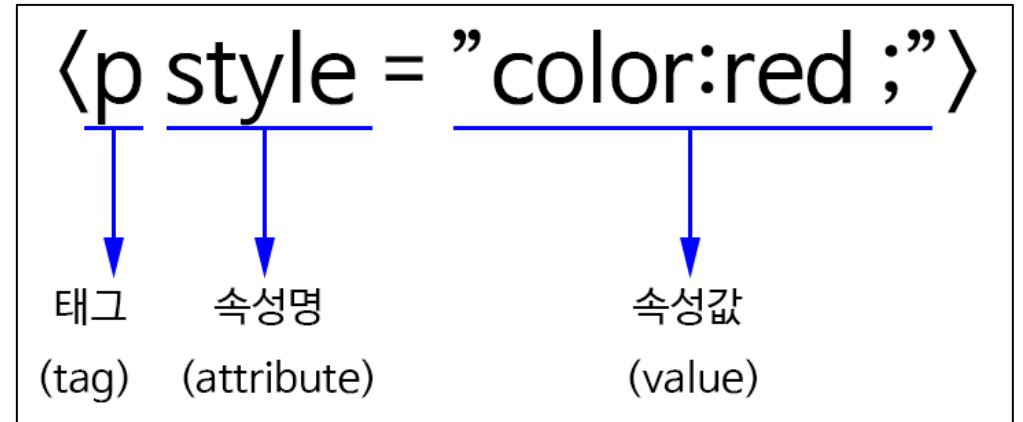
- 태그 내에 추가

- `style="속성 : 값 ; ..."`

- `<p style="color:blue; background-color: orange">내용</p>`

- `<h1 style="color:blue;">A Blue Heading</h1>`

- `<p style="color:red;">A red paragraph.</p>`

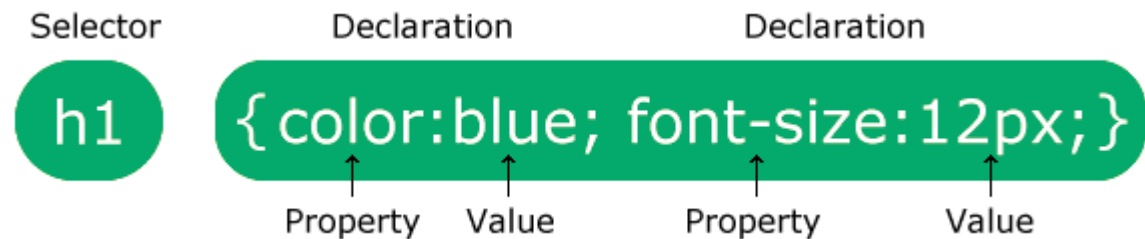


**A Blue Heading**

A red paragraph.

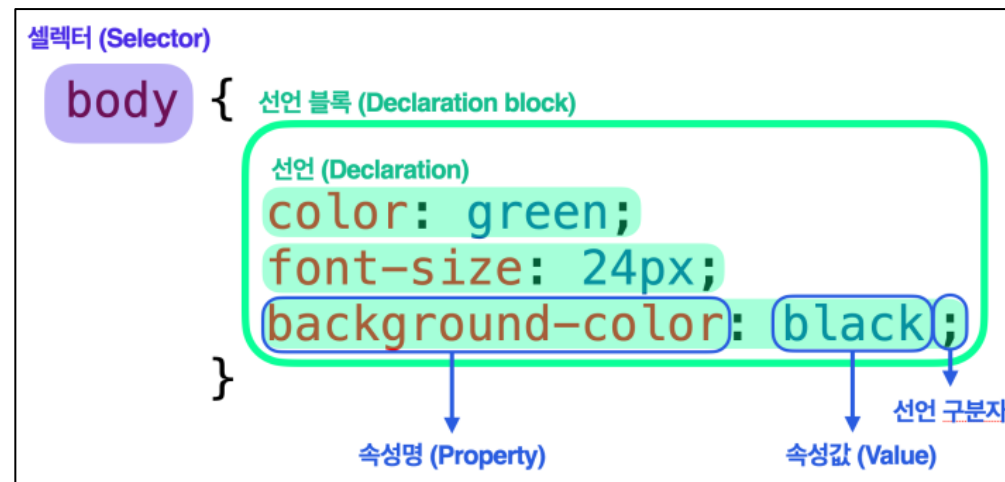
# CSS Syntax (internal, external)

- Selector
  - 단순 선택자 (tag, name, id, class)
  - 조합 선택자
  - Pseudo-class selectors (상태(state)에 따라 선택)
  - Pseudo-elements selectors (part of an element)
  - Attribute selectors (attribute or attribute value에 따라 선택)
- Declaration
  - 속성:값
  - 세미콜론으로 분리



# Internal CSS

```
<!DOCTYPE html>
<html>
<head>
<style>
body {background-color: powderblue;}
h1  {color: blue;}
p   {color: red;}
</style>
</head>
<body>
<h1>This is a heading</h1>
<p>This is a paragraph.</p>
</body>
</html>
```



**This is a heading**

This is a paragraph.

# Externam CSS

```
<!DOCTYPE html>
<html>
<head>
  <link rel="stylesheet" href="styles.css">
  <style>
    body {background-color: pink;}
  </style>
</head>
<body>
<h1>This is a heading</h1>
<p>This is a paragraph.</p>
</body>
</html>
```

**This is a heading**

This is a paragraph.

Style.css

```
body {
  background-color: powderblue;
}
h1 {  color: blue; }
p {  color: red; }
```

# 단순 선택자

Selector	Example	Example description
<u><a href="#">#id</a></u>	#firstname	Selects the element with id="firstname"
<u><a href="#">.class</a></u>	.intro	Selects all elements with class="intro"
<u><a href="#">element.class</a></u>	p.intro	Selects only <p> elements with class="intro"
<u><a href="#">*</a></u>	*	Selects all elements
<u><a href="#">element</a></u>	p	Selects all <p> elements
<u><a href="#">element,element,..</a></u>	div, p	Selects all <div> elements and all <p> elements

## Common CSS properties (by group)

### TEXT:

color  
font  
font-family  
font-size  
font-weight  
letter-spacing  
line-height  
text-align  
text-decoration  
text-indent  
text-transform  
vertical-align

### LIST:

list-style  
list-style-image  
list-style-position  
list-style-type

### BACKGROUND:

background  
background-attachment  
background-color  
background-image  
background-position  
background-repeat

### DISPLAY:

display  
float  
clear  
overflow  
visibility

### OTHER:

cursor

margin

border

padding

content

### BOX:

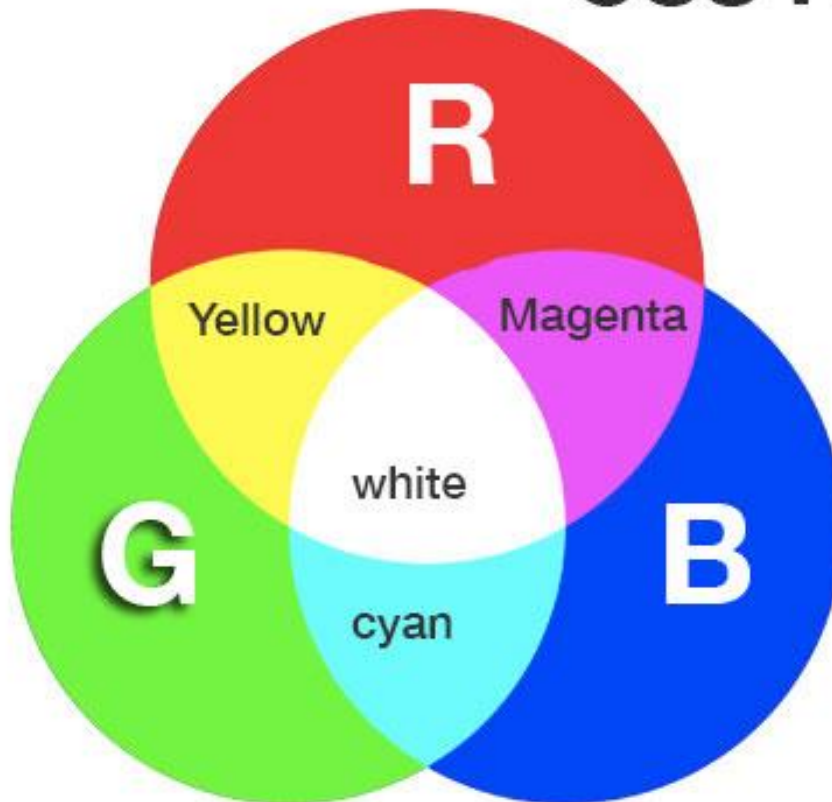
border  
border-color  
border-style  
border-width  
height  
margin  
padding  
width  
box-sizing

### POSITION:

position  
top  
bottom  
left  
right  
z-index

# CSS RGB Colors

## CSS RGB COLORS



**R** : #F00 or  
**RED** : #FF0000 or `rgb(255,0,0)`  
**G** : #00FF00 or `rgb(0,255,0)`  
**B** : #0000FF or `rgb(0,0,255)`

### HEXADECIMAL COLOR CODES

Red Color Code	Green Color Code	Blue Color Code
<b># RR</b>	<b>GG</b>	<b>BB</b>

— Signifies Hex Color Code



# Color CSS

```
<!DOCTYPE html>
<html>
<body>
<p>Same as color name "Tomato":</p>
<h1 style="background-color:rgb(255, 99, 71);">rgb(255, 99, 71)</h1>
<h1 style="background-color:#ff6347;">#ff6347</h1>
<h1 style="background-color:hsl(9, 100%, 64%);">hsl(9, 100%, 64%)</h1>
<p>Same as color name "Tomato", but 50% transparent:</p>
<h1 style="background-color:rgba(255, 99, 71, 0.5);">rgba(255, 99, 71, 0.5)</h1>
<h1 style="background-color:hsla(9, 100%, 64%, 0.5);">hsla(9, 100%, 64%, 0.5)</h1>
</body>
</html>
```

Same as color name "Tomato":

**rgb(255, 99, 71)**

**#ff6347**

**hsl(9, 100%, 64%)**

Same as color name "Tomato", but 50% transparent:

**rgba(255, 99, 71, 0.5)**

**hsla(9, 100%, 64%, 0.5)**

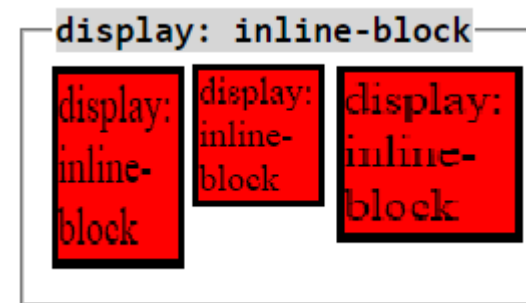
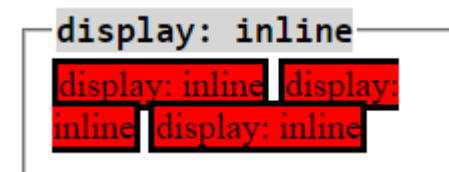
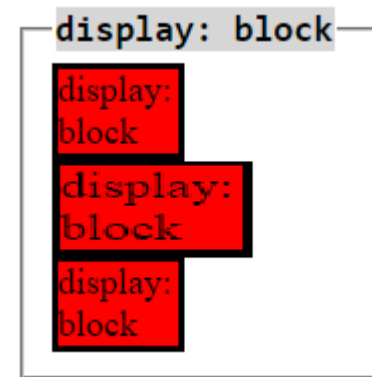
# Popular Color 이름

ivory	yellow	gold	orange	coral	tomato	red	maroon
lavender	pink	deep pink	plum	violet	fuchsia	purple	indigo
lime	green	olive	aqua	skyblue	teal	blue	navy
white	silver	gray		wheat	tan	chocolate	saddlebrown

# CSS Layout

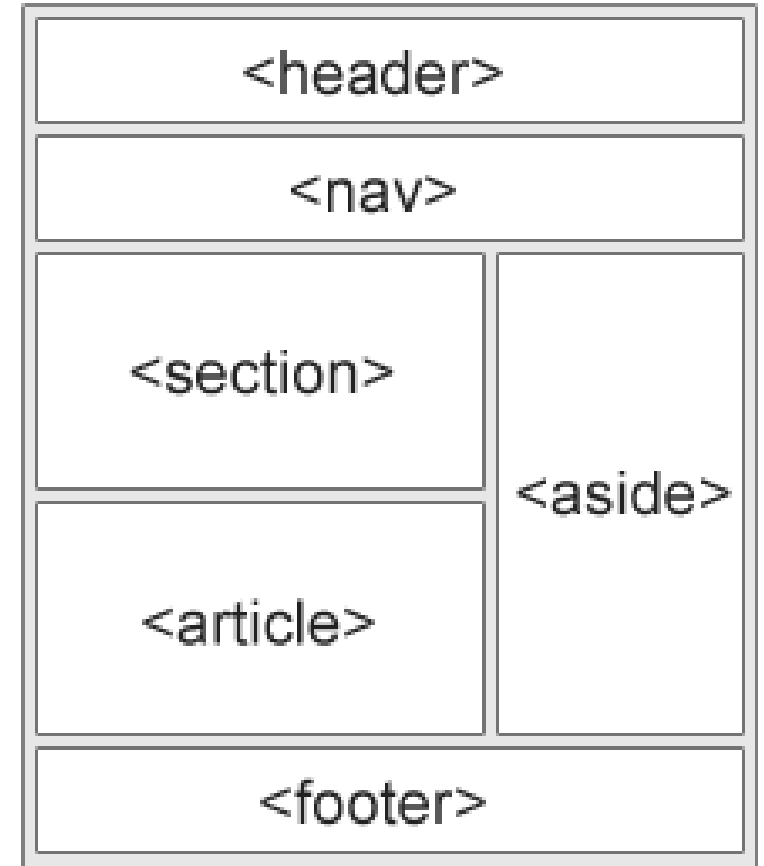
# HTML Basic Layout

- Normal flow
  - 차례로 배치
- Absolute/Relative
- CSS display 속성
  - block : div, h1
  - inline : span, img
  - inline-block
    - span { display: inline-block }



# HTML5 Layout Tags

- `<header>` defines a header for a document or a section
- `<nav>` defines a set of navigation links
- `<section>` defines a section in a document
- `<article>` defines an independent, self-contained content
- `<aside>` defines content aside from the content (like a sidebar)
- `<footer>` defines a footer for a document or a section
- `<details>` defines additional details that the user can open and close on demand
- `<summary>` defines a heading for the `<details>` element



# CSS 페이지 레이아웃 기술

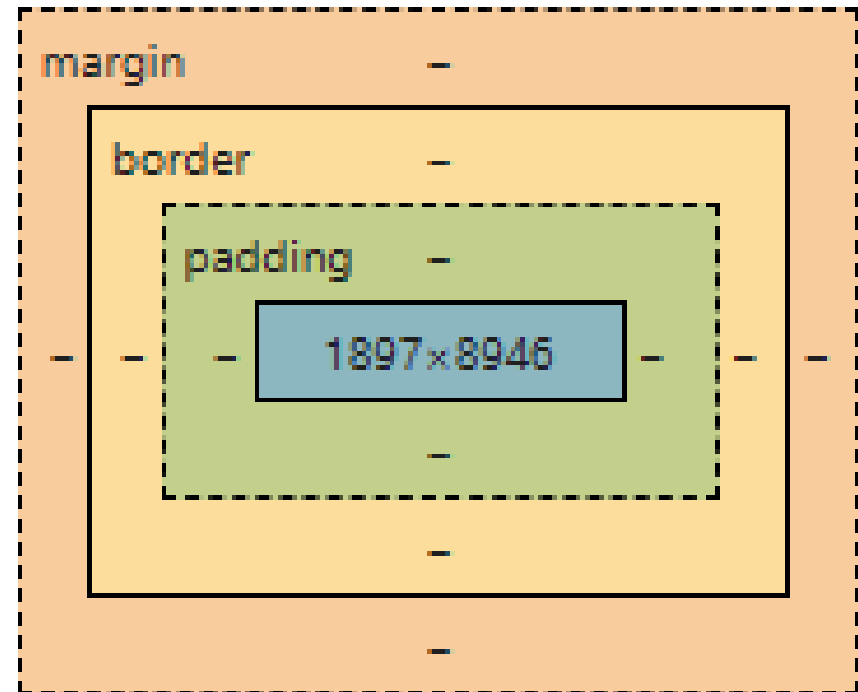
- 웹페이지에 포함될 요소들을 취합하여, 그들 요소가 일반 레이아웃 대열 상에 기본값 위치 기준과 부모 컨테이너, 또는 메인 뷰포인트 및 메인창과 비례해 어느 위치에 놓일 것인지를 제어
- 페이지 레이아웃 기술
  - 보통 흐름(normal flow)
  - display 속성
  - Flexbox, Grid, Floats
  - 포지셔닝
  - 테이블 레이아웃
  - 다단 레이아웃

# CSS Layout : position

- Static : default. 순서에 따라 그대로 배치
- Relative : 정상 위치에서 상대 변이
  - 기본값+상대적 위치
- Fixed : 스크롤이 되어도 화면의 고정 위치
  - 브라우저 화면의 좌상단을 기준으로 절대적인 위치
- Absolute : 부모 엘리먼트를 기준으로 절대 위치
- Sticky : 화면에 보일 때는 relative, 스크롤이 되어 안보이게 되면 경계에서 계속 보임
  - 기본적으로 relative처럼 작동
  - 스크롤 영역을 벗어나면 fixed처럼 작동
    - (ex)스크롤 내리면 따라다니는 팝업 광고창

# CSS Box Model

- Content
  - 콘텐츠
  - 크기, 색, 배경색 지정 가능
- Padding
  - 콘텐츠와 경계선 사이
  - 투명. 상하좌우 폭 지정 가능
- Border
  - 경계선
  - 색, 굵기 지정 가능
- Margin
  - 경계선 밖 인접 엘리먼트와의 간격
  - 투명. 상하좌우 폭 지정 가능







# CSS Flex(Flexbox) Layout

- flex CSS 속성
  - 하나의 플렉스 아이템이 자신의 컨테이너가 차지하는 공간에 맞추기 위해, 크기를 키우거나 줄이는 방법을 설정하는 속성
- Container에 display: flex 속성 지정

```
.abc { display: flex }
```

```
<div class=abc>  
  <div>AAA</div>  
  <div>BBBBBBBBBB</div>  
  <div>CCCCC</div>  
</div>
```



# CSS Layout : flex

- Container 내의 item 정렬

## CSS Flexbox Cheatsheet

**Justify Content**

- flex-start: 1 2 3
- flex-end: 1 2 3
- center: 1 2 3
- space-between: 1 2 3
- space-around: 3 2 1

**Flex Shrink**

- 0: 1
- 0.5: 1
- 1 or more: 1
- 0.25: 1 2
- 0.5 or more: 1 2

**Flex Wrap**

- no-wrap: 1 2 3
- wrap: 1 2 3
- wrap-reverse: 3 1 2

**Align Content**

- flex-start: 1 2 3
- flex-end: 1 2 3
- center: 1 2 3
- stretch: 1 2 3
- between: 1 2 3
- around: 1 2 3

**Align Self**

- stretch: 1 2 3 4
- flex-start: 2
- center: 3
- flex-end: 4

**Flex Direction**

- row: 1 2 3
- row-reverse: 3 2 1
- flex-start: 1 2 3
- flex-end: 1 2 3

**Flex Grow**

- 0 (0%): 1
- 0.5: 1
- 1 or more (100%): 1
- 0: 1 2
- 0.25: 1 2
- 0.5 or more: 2 3

@SuhailKakar

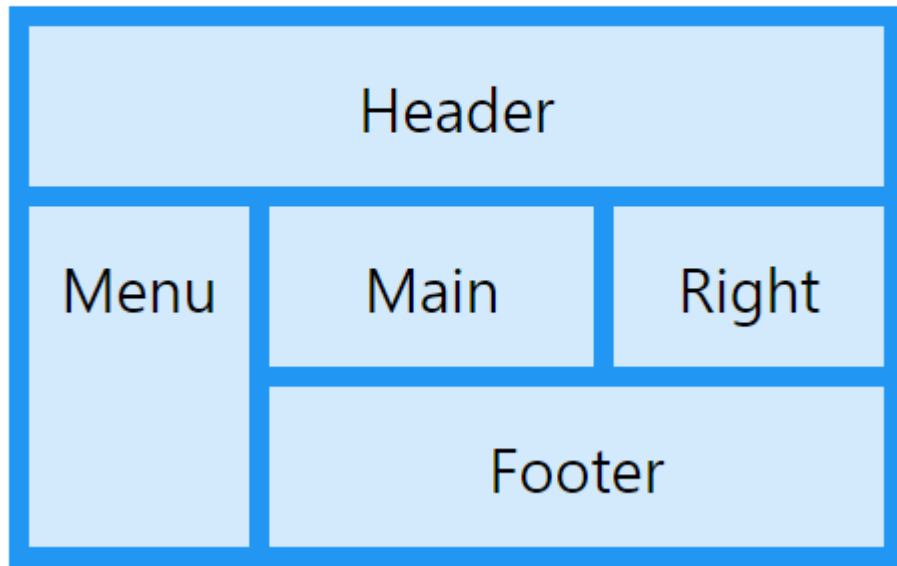
# CSS Grid

- CSS 그리드 레이아웃(Grid Layout)
  - 페이지를 여러 주요 영역으로 나누거나, 크기와 위치 및 문서 계층 구조의 관점에서 HTML 기본 요소로 작성된 컨트롤 간의 관계를 정의
  - 세로 열과 가로 행을 기준으로 요소를 정렬

```
.container {  
  display: grid;  
  grid-template-columns: 200px 200px 500px;  
  /* grid-template-columns: 1fr 1fr 1fr */ /* grid-template-columns: repeat(3, 1fr) */  
  /* grid-template-columns: 200px 1fr */ /* grid-template-columns: 100px 200px auto */  
  grid-template-rows: 200px 200px 500px;  
  /* grid-template-rows: 1fr 1fr 1fr */ /* grid-template-rows: repeat(3, 1fr) */  
  /* grid-template-rows: 200px 1fr */ /* grid-template-rows: 100px 200px auto */  
}
```

# Layout : grid

## Grid Layout



```
<!DOCTYPE html> <html> <head> <style>
.item1 { grid-area: header; } .item2 { grid-area: menu; }
.item3 { grid-area: main; }. item4 { grid-area: right; }
.item5 { grid-area: footer; }
.grid-container { display: grid;
  grid-template-areas:
    'header header header header header header'
    'menu main main main right right'
    'menu footer footer footer footer footer';
  gap: 10px;  background-color: #2196F3;
  padding: 10px; }
.grid-container > div {
  background-color: rgba(255, 255, 255, 0.8);
  text-align: center; padding: 20px 0; font-size: 30px; }
</style> </head> <body>
<h1>Grid Layout</h1>
<div class="grid-container">
  <div class="item1">Header</div>
  <div class="item2">Menu</div>
  <div class="item3">Main</div>
  <div class="item4">Right</div>
  <div class="item5">Footer</div>
</div> </body> </html>
```

# Layout : table 활용

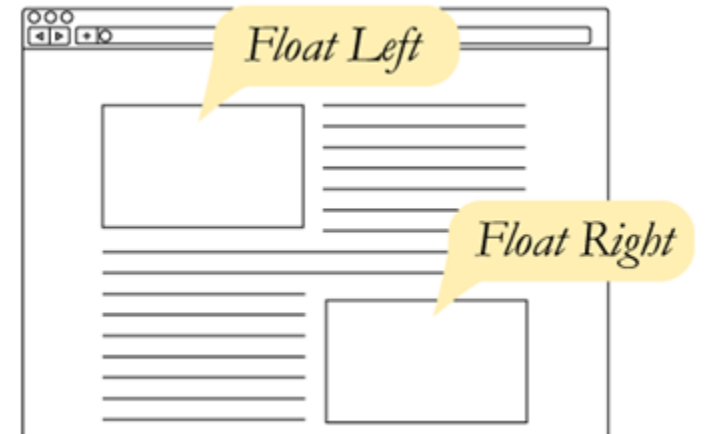
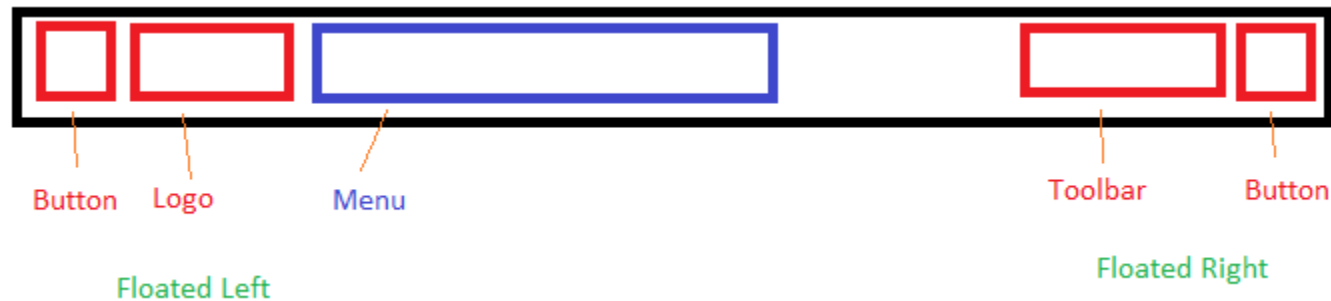
## Cell that spans two columns

Name		Age
xxx	yyy	12
aaa	bbb	

```
<!DOCTYPE html>
<html> <head>
<style>
table, th, td {
  border: 1px solid black;
  border-collapse: collapse;
  text-align : center
}
th { height: 80px; }
td { height: 50px; }
</style>
</head> <body>
<h2>Cell that spans two columns</h2>
<table style="width:100%">
  <tr> <th colspan="2">Name</th>
    <th>Age</th> </tr>
  <tr> <td>xxx</td> <td>yyy</td>
    <td rowspan="2">12</td> </tr>
  <tr> <td>aaa</td> <td>bbb</td> </tr>
</table>
</body>
</html>
```

# CSS Float Layout

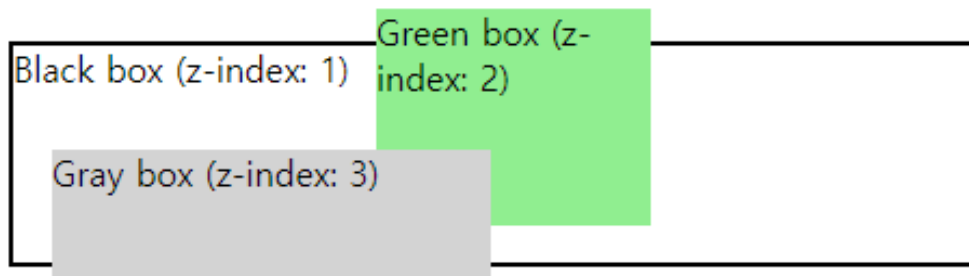
- CSS 속성(property) float
  - 한 요소(element)가 보통 흐름(normal flow)으로부터 빠져, 텍스트 및 인라인(inline) 요소가 그 주위를 감싸는 자기 컨테이너의 좌우 측을 따라 배치되어야 함을 지정
- Float 속성값 : left, right, none, inline-start, inline-end
- Float 지우기 : { clear: both }



# CSS Layout : z-index

- 엘리먼트가 겹칠 때 배치 순서 결정
- Default 0, + 혹은 - 값
- 같은 값이면 나중에 엘리먼트가 전면  
에 배치

## Z-index Example



```
<!DOCTYPE html> <html> <head> <style>
.container { position: relative; }
.black-box {
  position: relative;  z-index: 1;
  border: 2px solid black;
  height: 100px;  margin: 30px; }
.gray-box {
  position: absolute;  z-index: 3;
  background: lightgray;
  height: 60px;  width: 40%;
  left: 50px;  top: 50px; }
.green-box {
  position: absolute;  z-index: 2;
  background: lightgreen;
  width: 25%;  left: 200px;
  top: -15px;  height: 100px; }
</style> </head> <body>
<h1>Z-index Example</h1>
<div class="container">
  <div class="black-box">Black box (z-index: 1)</div>
  <div class="gray-box">Gray box (z-index: 3)</div>
  <div class="green-box">Green box (z-index: 2)</div>
</div> </body> </html>
```



# Block, inline, inline-block

- block element : div, h1..h6, p, ...
- inline element : span, img, a, ...
- display: inline-block
- display: inline
- display: block

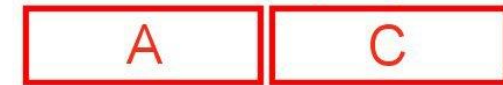
## Horizontal Navigation Links

[Home](#)[About Us](#)[Contact Us](#)

```
<!DOCTYPE html>
<html> <head> <style>
.nav { background-color: yellow;
      list-style-type: none; text-align: center;
      margin: 0; padding: 0;
    }
.nav li { display: inline-block;
          font-size: 20px; padding: 20px;
        }
</style> </head> <body>
<h1>Horizontal Navigation Links</h1>
<ul class="nav">
  <li> <a href="#home">Home</a> </li>
  <li> <a href="#about">About Us</a> </li>
  <li> <a href="#contact">Contact Us</a> </li>
</ul>
</body> </html>
```

# Display, visibility, opacity

- Display : block, inline, none(표시안함, 공간제거)
- Visibility : visible(보임), hidden(안보임,공간차지함), collapse(안보임, table과 flex에서는 공간도 제거)
- Opacity : 0(투명) – 1.0(default, 뒤를가림)



	Display: none;	Visibility: hidden;	Opacity: 0;
Collapses	✓	✗	✗
Events	✗	✗	✓
Taborder	✗	✗	✓
Animatable	✗	✗	✓

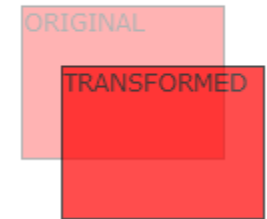
# CSS Animation

# CSS 2D Transforms

- move, rotate, scale, and skew elements
- 2D transformation methods
  - translate()
  - rotate()
  - scaleX()
  - scaleY()
  - scale()
  - skewX()
  - skewY()
  - skew()
  - matrix()

```
<style>
div {
  width: 300px; height: 100px;
  background-color: yellow;
  border: 1px solid black;
}
div {
  transform: translate(50px, 100px);
}
div#myDiv {
  transform: rotate(20deg);
}
</style>
```

## The translate()



## The rotate()



# CSS 3D Transforms

- CSS 3D Transforms Methods

- rotateX()
- rotateY()
- rotateZ()

## The rotateZ() Method

This a normal div element.

This div element is rotated  
90 degrees.

## The rotateZ() Method

This a nor

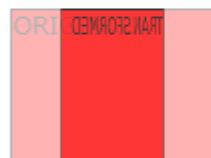
This div element is rotated  
90 degrees.

element.

## The rotateX()



## The rotateY()

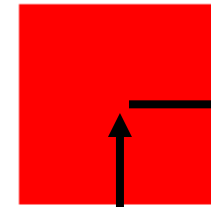


```
<!DOCTYPE html>
<html> <head> <style>
div {
  width: 200px;
  height: 50px;
  background-color: yellow;
  border: 1px solid black;
}
#myDivX {
  transform: rotateX(150deg);
}
#myDiv {
  transform: rotateZ(90deg);
}
</style> </head> <body>
<h2>The rotateZ() Method</h2>
<div>
This a normal div element.
</div>
<div id="myDiv">
This div element is rotated 90 degrees.
</div> </body> </html>
```

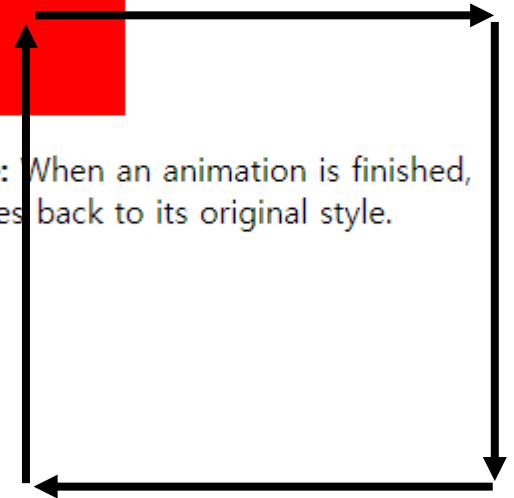
# CSS Animation

```
<!DOCTYPE html> <html> <head> <style>
div { width: 100px; height: 100px; background-color: red;
  position: relative; animation-name: example;
  animation-duration: 4s; }
@keyframes example {
  0%   {background-color:red; left:0px; top:0px;}
  25%  {background-color:yellow; left:200px; top:0px;}
  50%  {background-color:blue; left:200px; top:200px;}
  75%  {background-color:green; left:0px; top:200px;}
  100% {background-color:red; left:0px; top:0px;} }
</style> </head> <body>
<h1>CSS Animation</h1>
<div> </div>
<p> <b>Note:</b> When an animation is finished,<br>
it goes back to its original style.</p>
</body> </html>
```

## CSS Animation



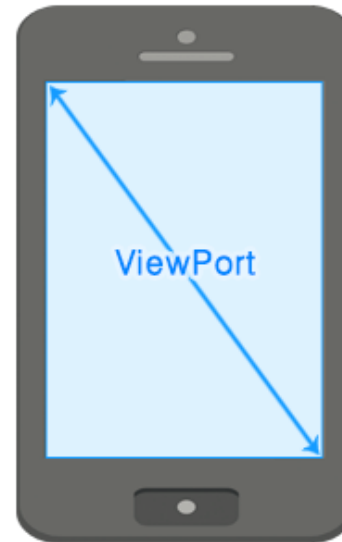
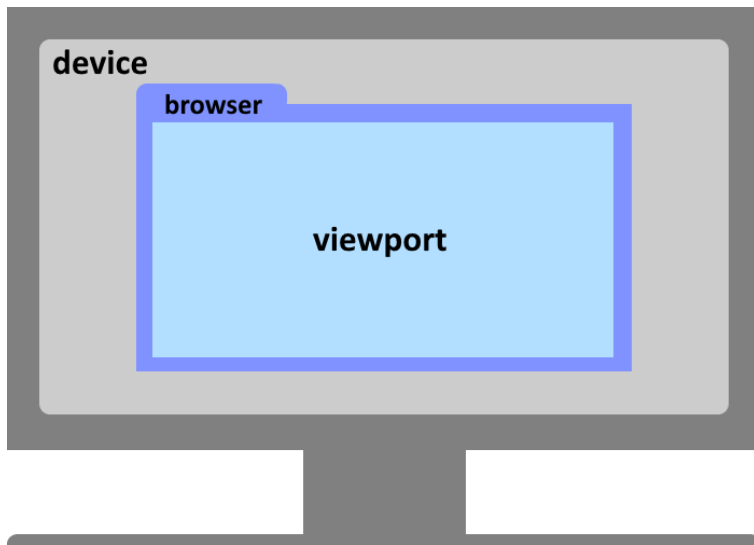
**Note:** When an animation is finished, it goes back to its original style.



# Viewport, Media query

# Viewport

- 현재 보고 있는 컴퓨터 화면의 영역
  - 콘텐츠가 표시되는 부분
  - 브라우저에서 UI, 메뉴바 등을 제외한 창
- 대형 모니터에서 뷰포트는 브라우저 창의 크기
- 모바일 디바이스에서 뷰포트는 브라우저 전체 화면





# View port 설정 for mobile

- HEAD에서 meta 태그를 이용하여 설정
- `<meta name="viewport" content="width=device-width, initial-scale=1.0", user-scalable=no, minimum-scale=1.0, maximum-scale=2.0">`



# Media query

- 콘텐츠 렌더링이 화면 해상도와 같은 다양한 조건에 적응할 수 있도록 하는 CSS 3의 기능
- HEAD에서 정의
  - @media (조건문) { 실행코드 }
  - @media media-type and (expression) {  
    <style css>  
    }
  - media-type : screen, print, speech, all
- 예제

```
@media (max-width: 750px) {  
  p {  
    font-size: 14px;  
  }  
}
```

# Media query example

```
@media only screen and  
(max-width: 600px) {  
  div.example {  
    display: none;  
  }  
}
```

```
<link rel="stylesheet"  
media="screen and (max-  
width: 768px)"  
href="mystyle.css" />
```

```
#media-320, #media-768, #media-1024 {  
  display: none;  
  height: 0px;  
  overflow: hidden;  
}  
@media all and (max-width: 320px) {  
  #media-320 { display: block; }  
}  
@media all and (min-width: 321px) and (max-width: 768px) {  
  #media-768 { display: block; }  
}  
@media all and (min-width: 769px) and (max-width: 1024px) {  
  #media-1024 { display: block; }  
}
```