

HTML



Markup Language
Content

CSS



Style sheet Language
Presentation

JS



Programming Language
Behavior

CSS 레이아웃

<https://www.w3schools.com/html/default.asp>

레이아웃(Layout)

- 웹 사이트를 구성하는 요소들을 배치할 공간을 분할하고 정렬

- float 속성

- 요소를 왼쪽이나 오른쪽에 떠 있게 만들
- 기본형
 - float: left | right | none

```
<style>
  img {
    float: left;
    margin-right: 50px;
    margin-bottom: 30px;
    width: 200px;
  }
</style>
</head>
<body>
  
  <p>
    Lorem ipsum dolor sit amet, consecte
  </p>
```

- clear 속성

- float 속성을 무효화 시키는 속성
- 기본형
 - clear: left | right | none | both



Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed
eiusmod tempor incididunt ut labore et dolore magna alio
enim ad minim veniam, quis nostrud exercitation ullamco
nisi ut aliquip ex ea commodo consequat. Duis aute irure
reprehenderit in voluptate velit esse cillum dolore eu fugia
pariatur. Excepteur sint occaecat cupidatat non proident, s
culpa qui officia deserunt mollit anim id est laborum.

예제> 레이아웃 만들기

```
<header>
  <h1>웹 프로그래밍 개발</h1>
  <nav> 상단 메뉴 구성 </nav>
</header>
<div id="content">
  <aside id="left_menu">
    왼쪽 메뉴 구성
  </aside>
  <section>
    <article>내용1</article>
    <article>내용2</article>
    <article>내용3</article>
  </section>
  <aside id="adv">우측 광고 </aside>
</div>
<footer>
  카피라이트
</footer>
```

웹 프로그래밍 개발

상단 메뉴 구성
왼쪽 메뉴 구성
내용1
내용2
내용3
우측 광고
카피라이트

웹 프로그래밍 개발

왼쪽 메뉴 구성

내용1
내용2
내용3

상단 메뉴 구성

우측 광고

카피라이트

예제> 레이아웃 만들기(계속)

```
<header>
  <h1>웹 프로그래밍 개발</h1>
  <nav> 상단 메뉴 구성 </nav>
</header>
<div id="content">
  <aside id="left_menu">
    왼쪽 메뉴 구성
  </aside>
  <section>
    <article>내용1</article>
    <article>내용2</article>
    <article>내용3</article>
  </section>
  <aside id="adv">우측 광고 </aside>
</div>
<footer>
  카피라이트
</footer>
```

웹 프로그래밍 개발

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

메뉴

HTML
CSS
JavaScript

HTML

하이퍼 텍스트 마크업 언어(Hyper Text Markup Language, HTML)는 웹 페이지 표시를 위해 개발된 지배적인 마크업 언어다. 또한, HTML은 제목, 단락, 목록 등과 같은 본문을 위한 구조적 의미를 나타내는 것뿐만 아니라 링크, 인용과 그 밖의 항목으로 구조적 문서를 만들 수 있는 방법을 제공한다. 그리고 이미지와 객체를 내장하여 대화형 양식을 생성하는 데 사용될 수 있다. HTML은 웹 페이지 콘텐츠 안의 각 소괄호에 둘러싸인 "태그"로 되어있는 HTML 요소 형태로 작성한다. HTML은 웹 브라우저와 같은 HTML 처리 장치의 행동에 영향을 주는 자바스크립트, 본문과 그 밖의 항목의 외관과 배치를 정의하는 CSS 같은 스크립트를 포함하거나 불러올 수 있다. HTML과 CSS 표준의 공동 책임자인 W3C는 명확하고 표상적인 마크업을 위하여 CSS의 사용을 권장한다.

CSS

캐스케이딩 스타일 시트(Cascading Style Sheet)는 마크업 언어가 실제 표시되는 방법을 기술하는 스타일 언어(Style sheet language)로, HTML과 XHTML에 주로 쓰이며, XML에서도 사용할 수 있다. W3C의 표준이고, 레이아웃과 스타일을 정의할 때의 자유도가 높다.

마크업 언어(ex: HTML)가 웹사이트의 몸체를 담당한다면 CSS는 옷과 액세서리처럼 꾸미는 역할을 담당한다고 할 수 있다. 즉 HTML 구조는 그대로 두고

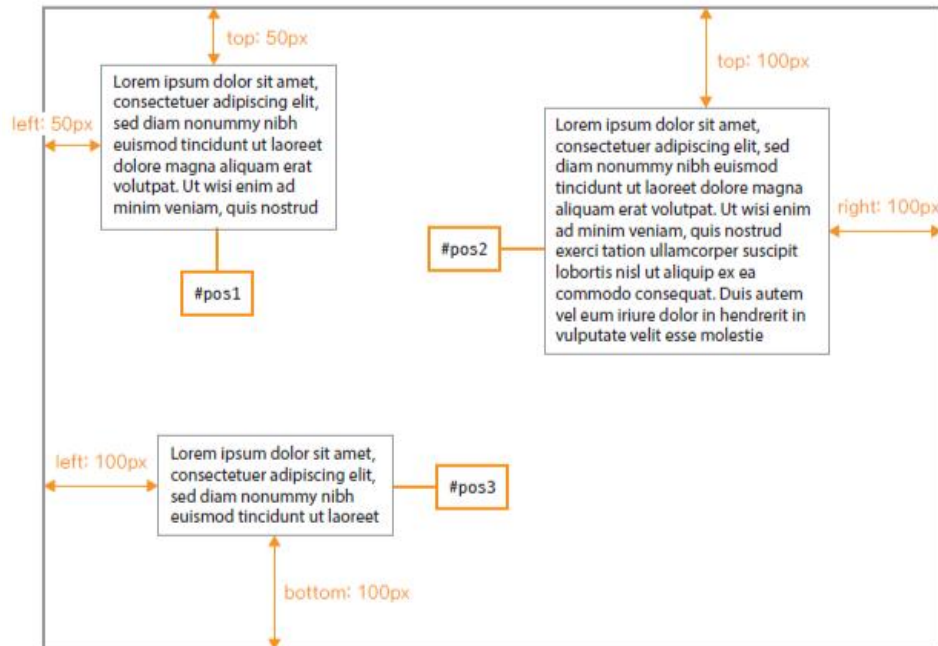
광고

광고 내용

웹 요소의 위치 지정하기

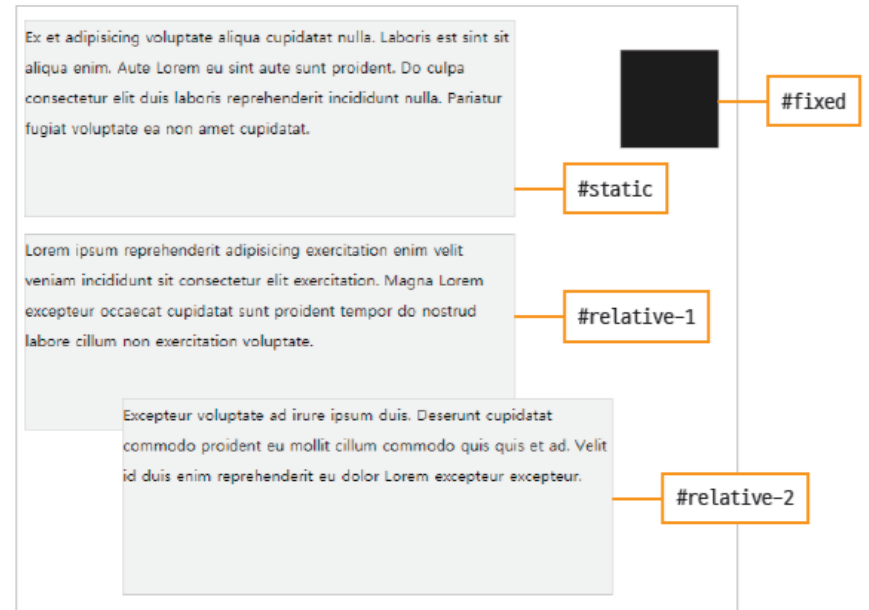
■ 웹 요소의 위치를 지정하는 속성

- left, right, bottom, top 속성
 - 예: left: 50px;
 - 기준 위치와 요소 사이에 왼쪽으로 50px 만큼 떨어진 위치로 지정



■ position 속성

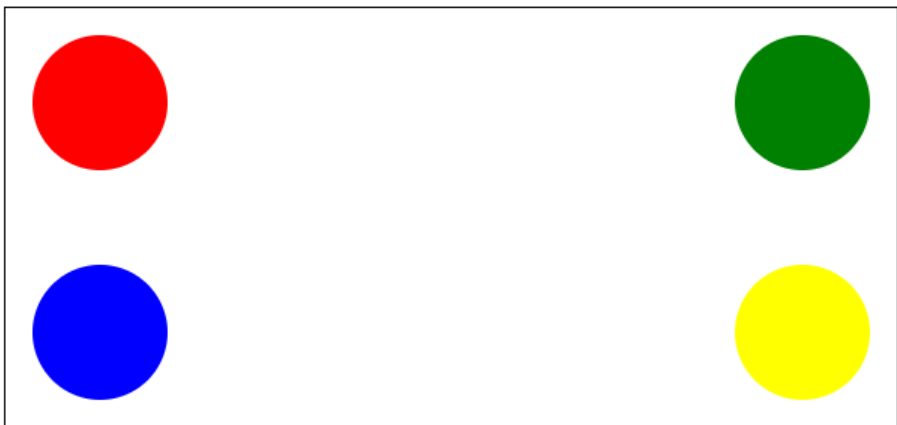
- 웹 문서 안에 요소들을 배치하기 위한 속성
- 기본형
 - position: static | relative | absolute | fixed



웹 요소의 위치 지정하기 예제

```
<h1>Circle </h1>
<div id="container">
  <div id="red" class="circle"></div>
  <div id="green" class="circle"></div>
  <div id="blue" class="circle"></div>
  <div id="yellow" class="circle"></div>
</div>
<h1>절대 위치 사용 예제</h1>
```

Circle



절대 위치 사용 예제

```
<style>
#container {
  width: 650px; height: 300px;
  border: 2px solid black;
  padding: 5px;
  position: relative;
}
.circle{
  width: 100px; height: 100px;
  border-radius: 50%;
  position: absolute;
}
#red{
  background-color: red;
  top: 20px;
  left: 20px;
}
#green{
  background-color: green;
  top: 20px;
  right: 20px;
}
#blue{
  background-color: blue;
  bottom: 20px;
  left: 20px;
}
#yellow{
  background-color: yellow;
  bottom: 20px;
  right: 20px;
}
</style>
```

자손의 *position* 속성에
*absolute*를 적용하려면
부모의 *position* 속성에
*relative*를 적용

요소를 중앙에 배치

■ 방법

- 중앙 정렬하려는 div 태그의 position 속성을 absolute로 지정
- left 속성과 top 속성을 모두 50%로 지정
- 중앙에 정렬하려는 div 태그의 margin-left, margin-top 속성에 음수를 입력

```
<body>
  <div id="container">
    <h1> 요소의 중앙 배치</h1>
  </div>
```



```
*{
  margin: 0;
  padding: 0;
}
body{
  background-color: #f0f0f0;
}
#container {
  width: 400px; height: 200px;
  border: 2px solid black;
  background: yellow;
  position: absolute;
  left: 50%;
  top: 50%;
  margin-left: -200px;
  margin-top: -100px;
  text-align: center;
  line-height: 200px;
}
```

반응형 웹

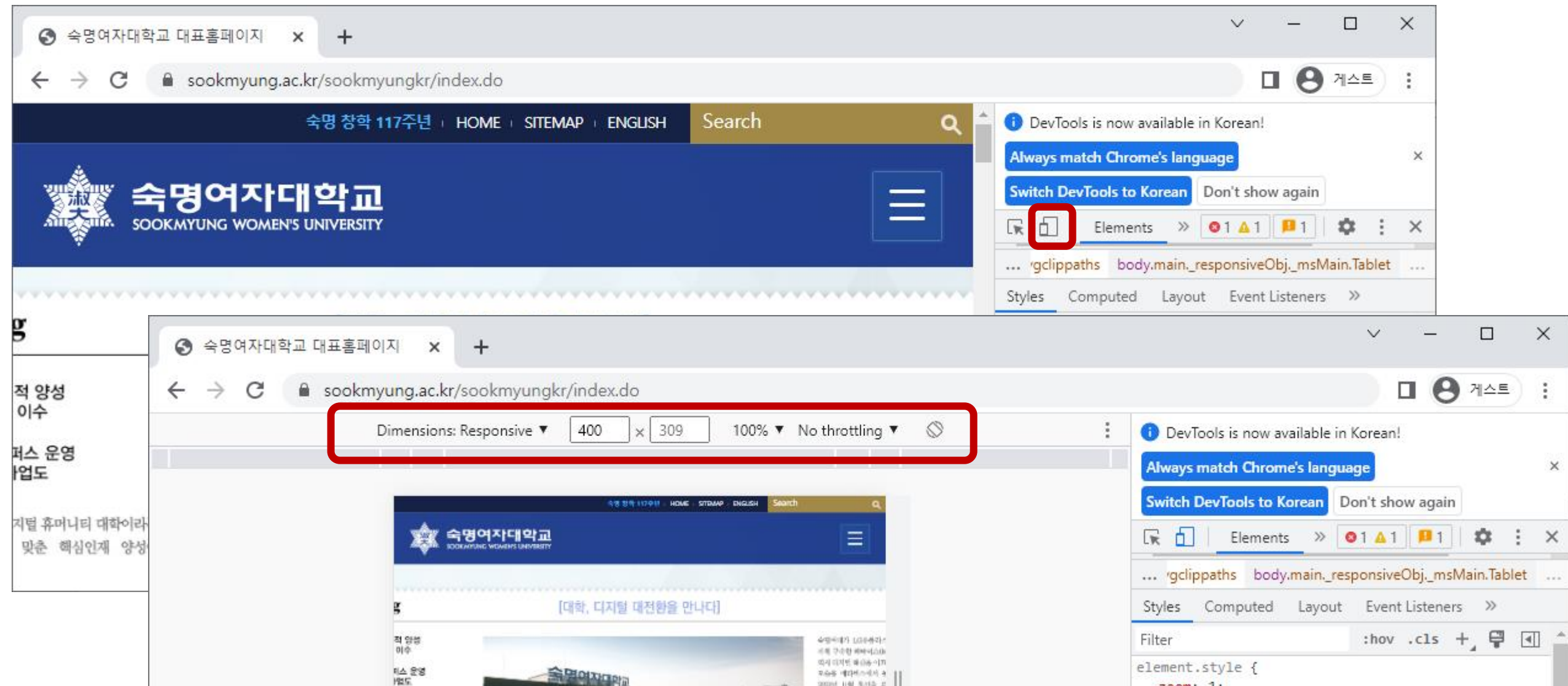
- 하나의 웹 사이트의 내용을 그대로 유지하면서 다양한 디바이스(데스크톱, 태블릿PC, 스마트폰)의 화면 해상도에 맞게 웹 사이트를 표시하는 방법
- 모바일 기기를 위한 뷰포트(viewport)
 - 뷰포트: 실제 내용이 표시되는 영역
 - 뷰포트 지정하기

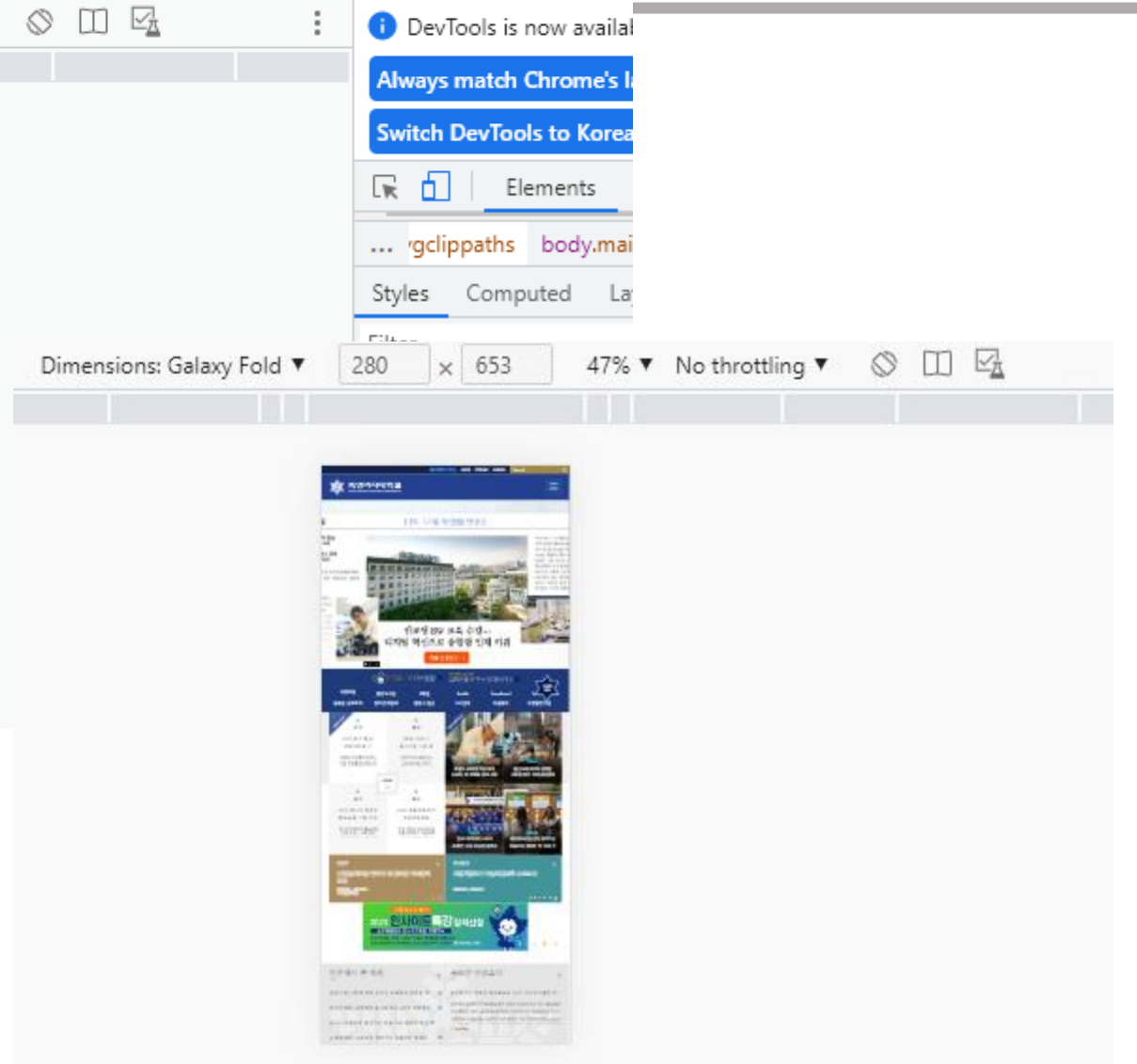
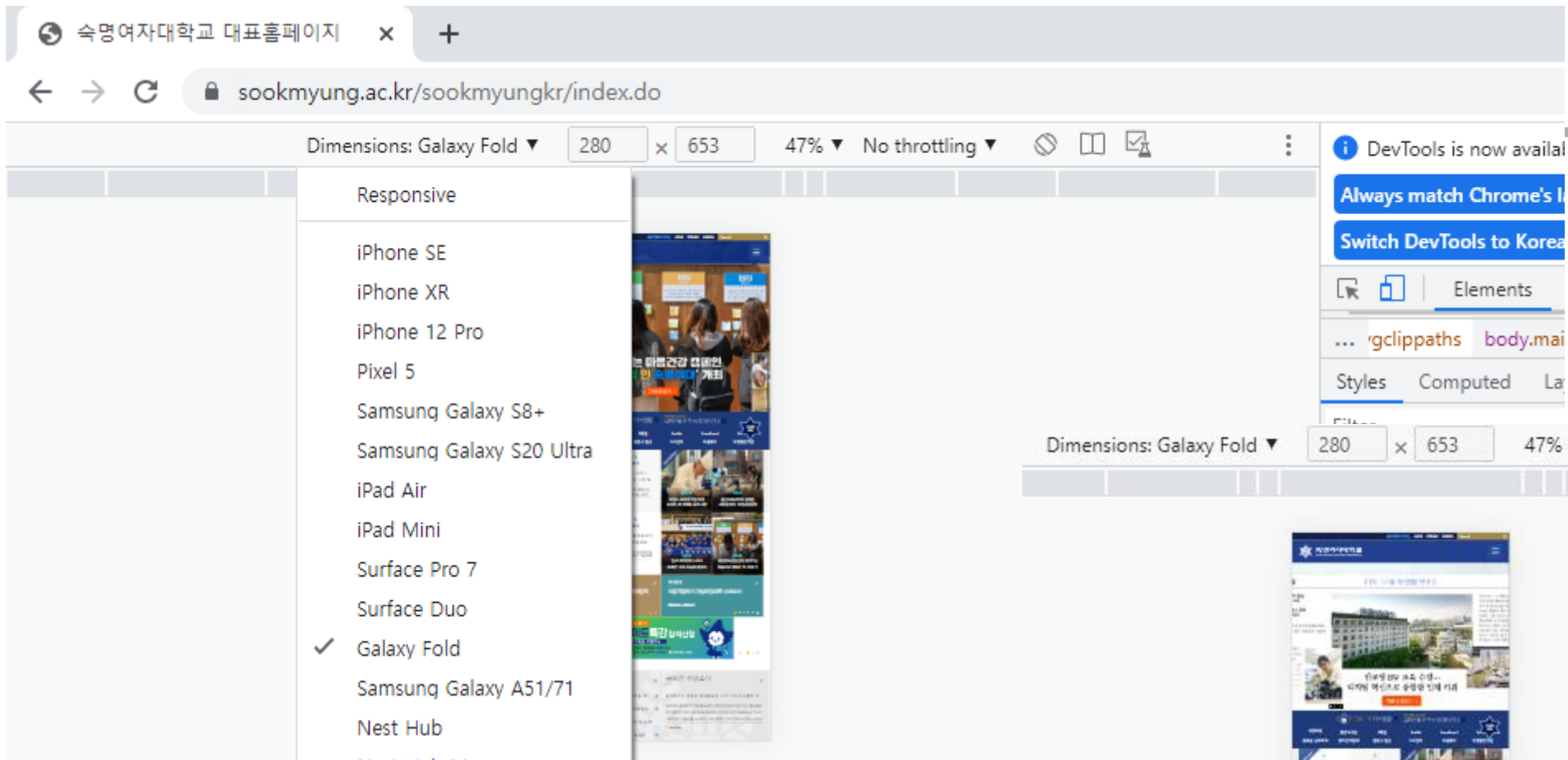
```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

 - 너비를 디바이스 너비에 맞추고 초기 화면 배율을 1로 지정
 - 뷰포트 단위
 - vw(viewport width): 1vw는 뷰포트 너비의 1%
 - vh(viewport height): 1vh는 뷰포트 높이의 1%
 - vmin, vmax: 뷰포트 너비와 높이 중에서 최소/최대값의 1%

크롬 브라우저의 디바이스 모드 활용하기

■ 크롬 개발자 도구의 디바이스 모드 활용





미디어 쿼리(Media Queries)

- 접속하는 장치(미디어)에 따라 특정한 css 스타일을 사용하는 방법

- 미디어 쿼리 구문

- @media 속성을 사용해 특정 미디어에서 어떤 css를 적용할 것인지 지정
- `@media screen and (min-width: 768px) and (max-width: 1024px){`
`/* 스타일 지정 */`
`}`
@media [only | not] 미디어 유형 {and 조건}

- 미디어 유형의 종류

- all, print, screen, tv, handheld, ...

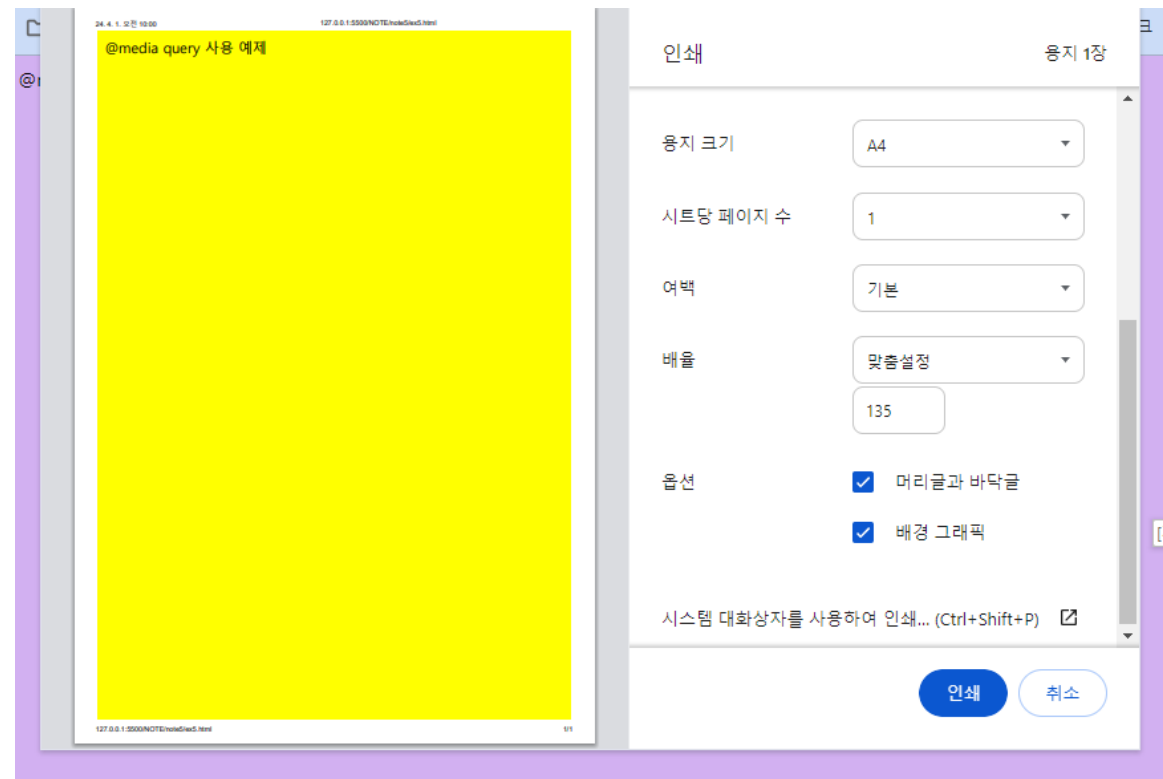
@media 규칙을 사용한 미디어 쿼리 예제

@media query 사용 예제

```
<!DOCTYPE html>
<html lang="en">
<head>
  <style>
    @media screen {
      body {background-color: #14181c;rgb(210, 176, 241);}
    }
    @media print {
      body {background-color: #ffff00;yellow;}
    }
  </style>
</head>
<body>
  @media query 사용 예제
</body>
</html>
```

■ media 속성을 사용한 미디어 쿼리 사용

```
<link rel="stylesheet" href="screen.css" media="screen">
<link rel="stylesheet" href="print.css" media="print">
```



미디어 쿼리의 다양한 조건

모바일 기기의 뷰포트 크기 확인
<https://yesviz.com/devices.php>

■ 웹 문서의 가로 너비와 세로 높이(뷰포트)

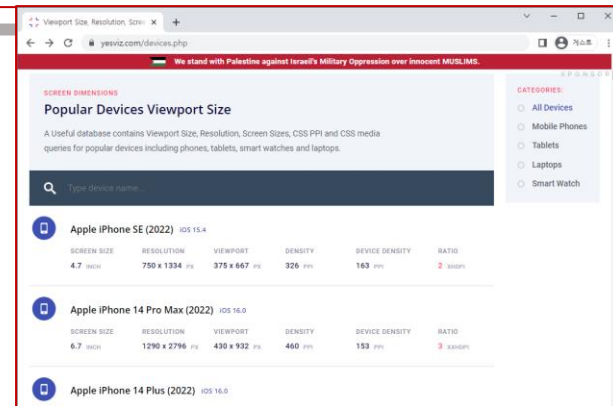
- 실제 웹 문서 내용이 나타나는 영역의 너비와 높이를 조건으로 사용
- width, height, min-width, min-height, max-width, max-height

■ 단말기의 가로 너비와 세로 높이

- device-width, device-height
- min-device-width, min-device-height, max-device-width, max-device-height

■ 화면 회전

- orientation: portrait, orientation: landscape



The screenshot shows the 'Popular Devices Viewport Size' page on yesviz.com. It features a search bar and a table of device specifications. The table has columns for Screen Size, Resolution, Viewport, Density, Device Density, and Ratio. Three devices are listed: Apple iPhone SE (2022), Apple iPhone 14 Pro Max (2022), and Apple iPhone 14 Plus (2022).

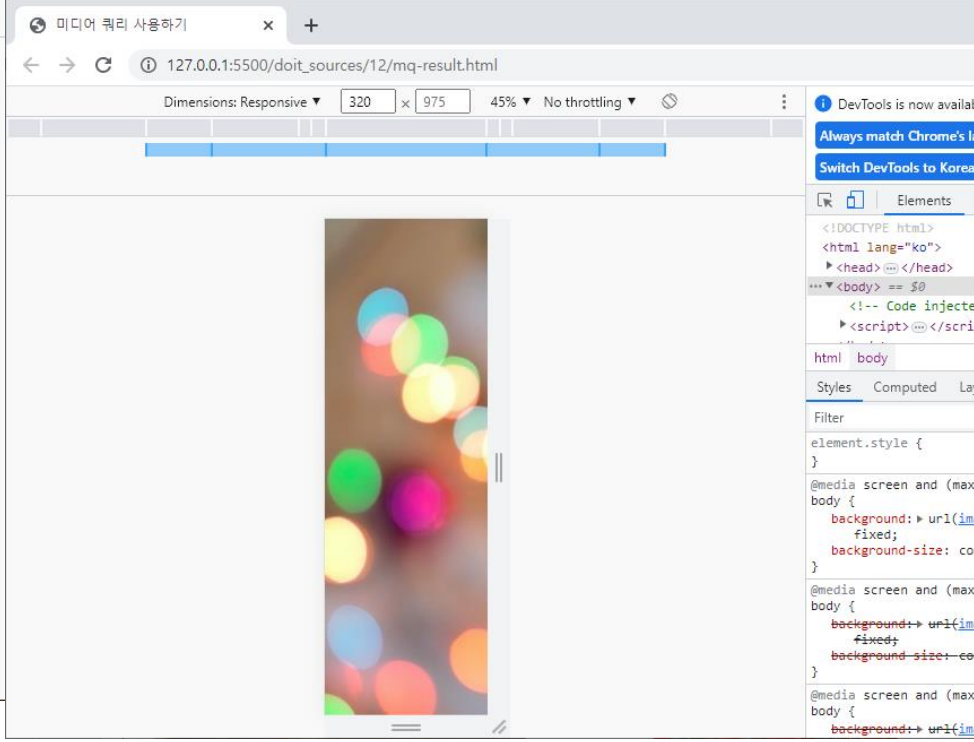
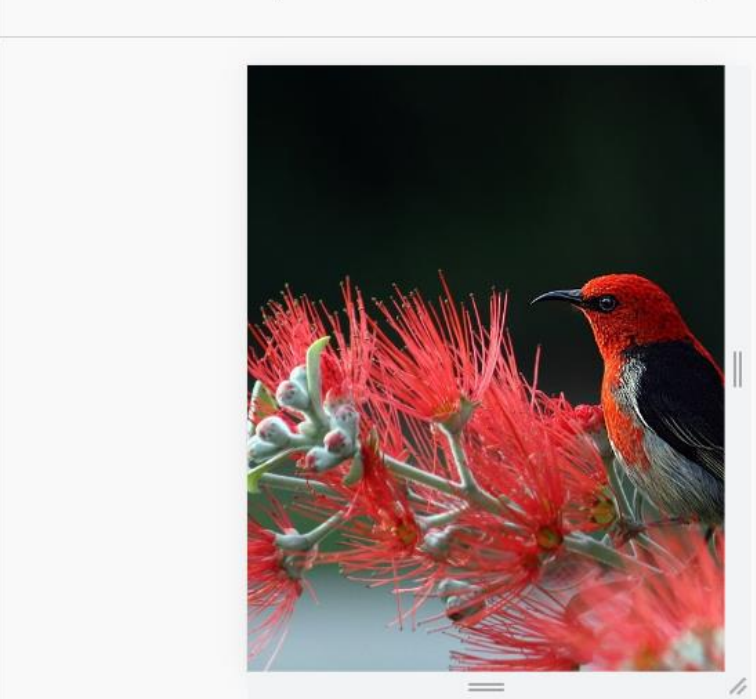
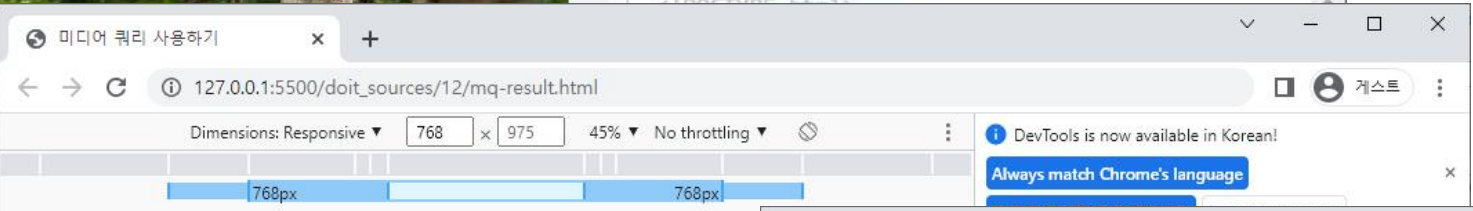
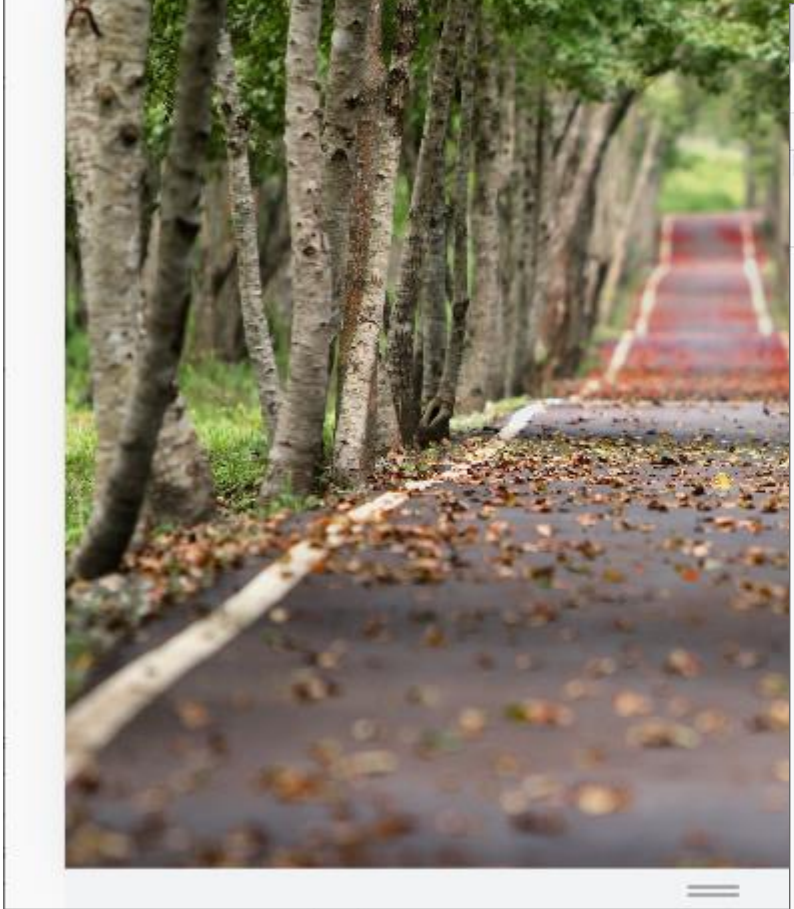
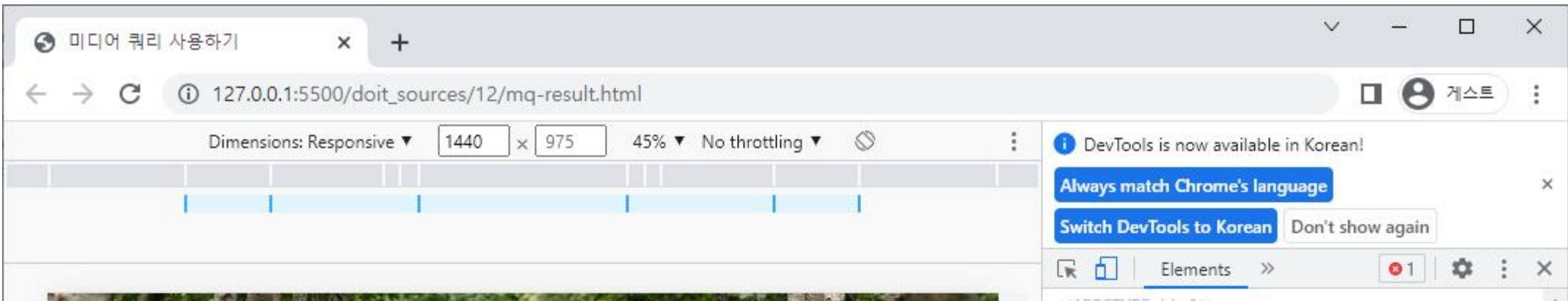
SCREEN SIZE	RESOLUTION	VIEWPORT	DENSITY	DEVICE DENSITY	RATIO
4.7 inch	750 x 1334 px	375 x 667 px	326 ppi	163 ppi	2 x@1px
6.7 inch	1290 x 2796 px	430 x 932 px	460 ppi	153 ppi	3 x@1px
6.7 inch	1290 x 2796 px	430 x 932 px	460 ppi	153 ppi	3 x@1px

스마트폰(480),
태블릿(768),
데스크탑

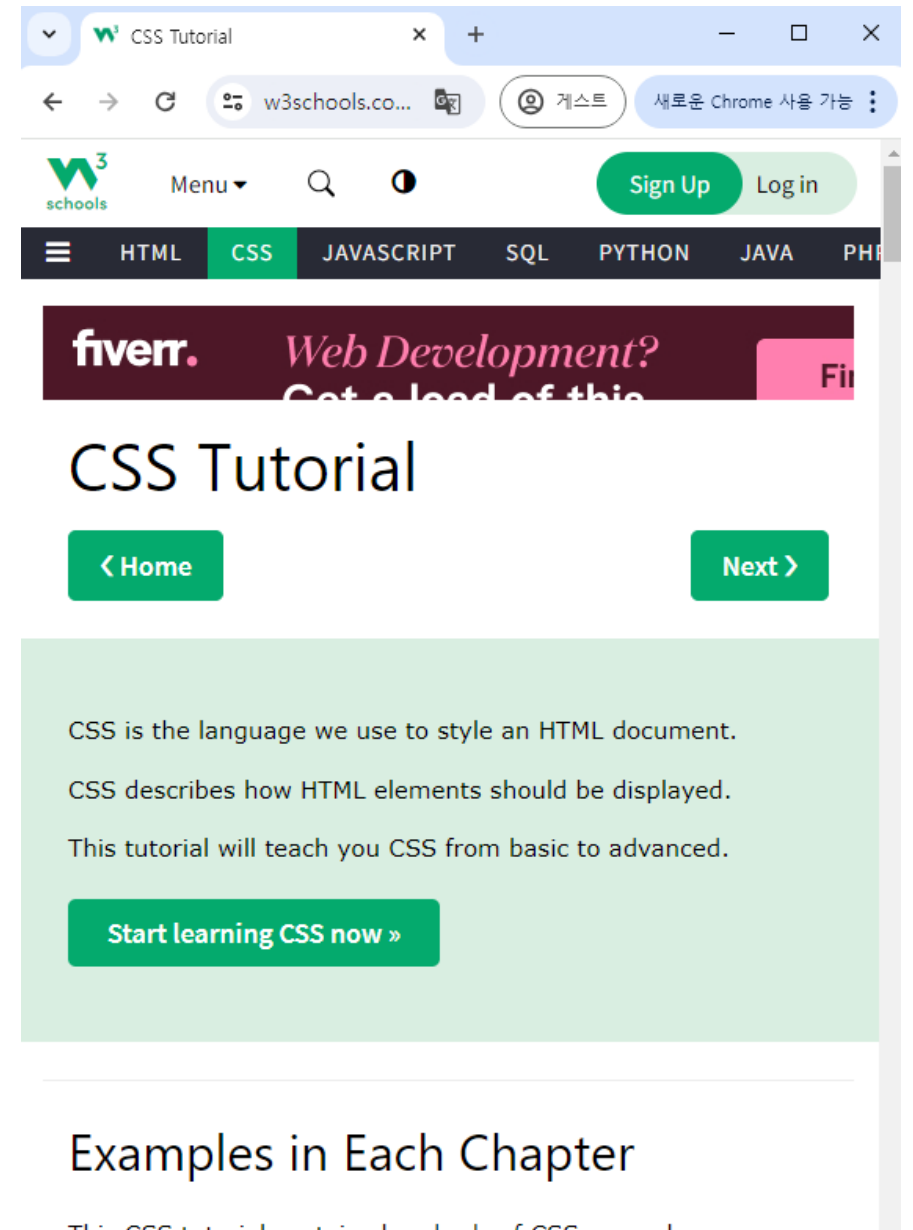
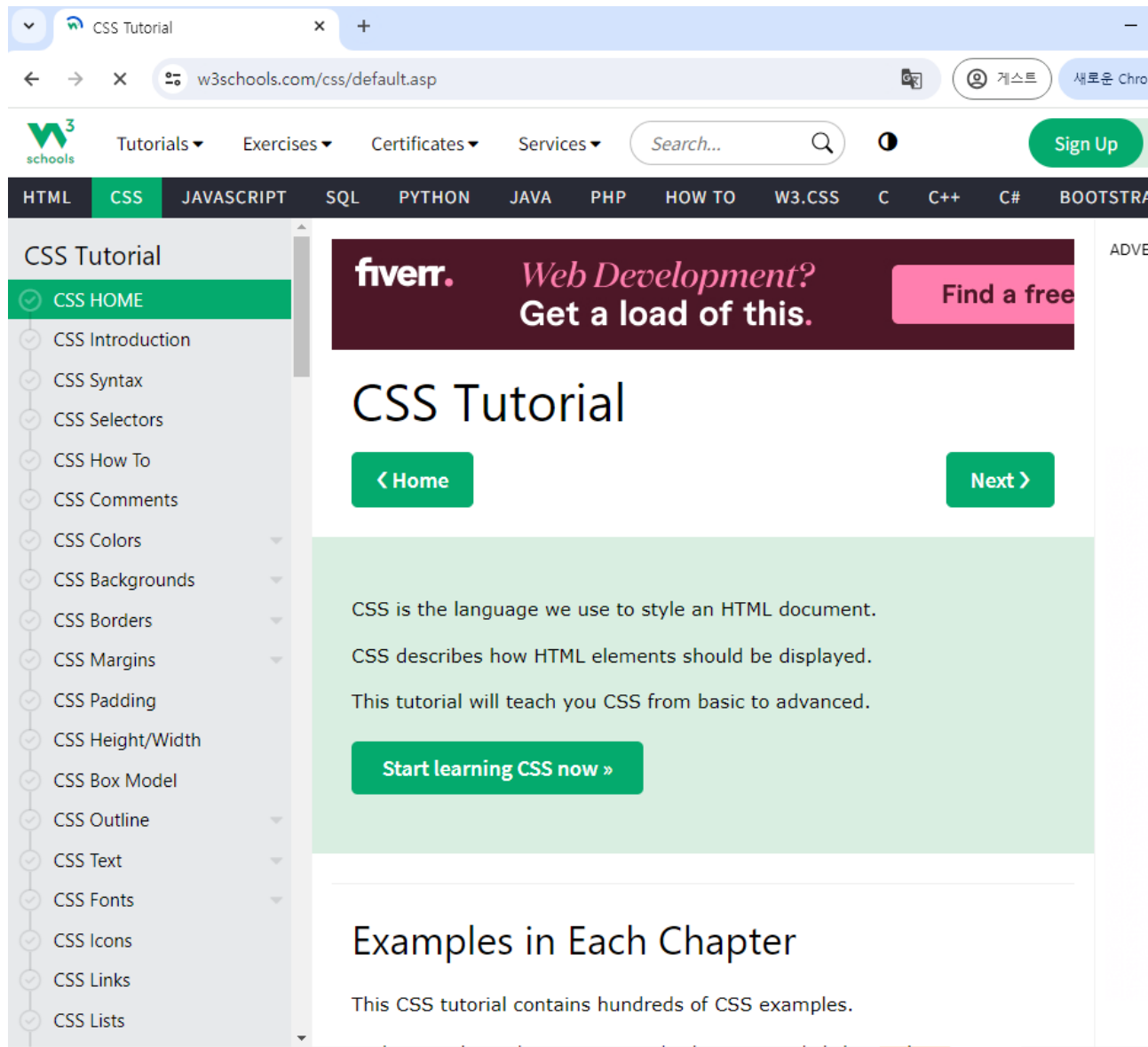
```
<link rel="stylesheet" href="phone.css" media="screen and (max-width: 767px)">  
<link rel="stylesheet" href="tablet.css" media="screen and (min-width: 768px) and (max-width: 959px)">  
<link rel="stylesheet" href="desktop.css" media="screen and (min-width: 960px)">
```

화면 크기에 따라 배경 이미지 바꾸기

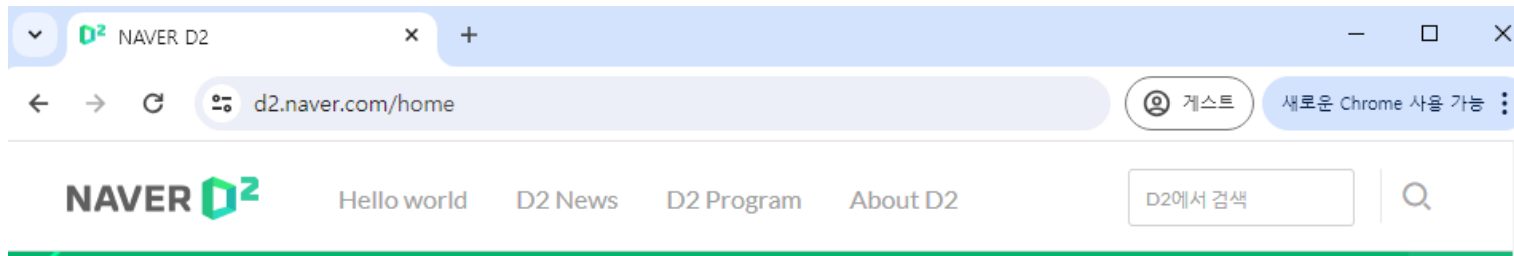
```
<style>
  body {
    background: url(images/bg0.jpg) no-repeat fixed; /* 기본 배경 이미지 지정 */
    background-size: cover;
  }
  @media screen and (max-width:1024px) {
    body {
      background: url(images/bg1.jpg) no-repeat fixed; /* 가로가 1024px 이하면 bg1.jpg */
      background-size: cover;
    }
  }
  @media screen and (max-width:768px) {
    body {
      background: url(images/bg2.jpg) no-repeat fixed; /* 가로가 768px 이하면 bg2.jpg */
      background-size: cover;
    }
  }
  @media screen and (max-width:320px) {
    body {
      background: url(images/bg3.jpg) no-repeat fixed; /* 가로가 320px 이하면 bg3.jpg */
      background-size: cover;
    }
  }
</style>
```

반응형 웹 패턴(<https://www.w3schools.com/css/default.asp>)



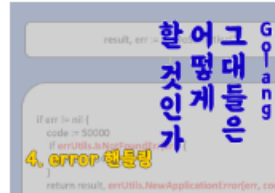
반응형 웹 패턴(http://d2.naver.com/home)



Golang, 그대들은 어떻게 할 것인가 - 4. error 핸들링

앞 글에서는 하위 레이어에서 error를 어떻게 반환할지에 대해 다뤘습니다. 이번 글에서는 반환받은 error를 상위 함수에서는

2024.03.29 | 577



Golang, 그대들은 어떻게 할 것인가 - 3. error 래핑

앞 글에서는 각 서버에 산재해 있던 MongoDB 관련 코드를 공통화 및 추상화하고, DB error 정보를 담을 수 있는 구조를 만

2024.03.29 | 499



Golang, 그대들은 어떻게 할 것인가 - 2. MongoDB Go Driver 추상화

클로바노트 V1의 주요 서버들은 Golang(v1.14)으로 개발되었고 MongoDB를 메인 DB로 사용하고 있습니다. MongoDB



TOP 5

- 1 일 3,000만 건의 네이버페이 주문 메...
- 2 Virtual Thread의 기본 개념 이해하기
- 3 Golang, 그대들은 어떻게 할 것인가 - ...
- 4 Golang, 그대들은 어떻게 할 것인가 - ...
- 5 백엔드 개발자를 꿈꾸는 학생개발자...

TOP KEYWORDS

빅데이터

분산처리

Golang

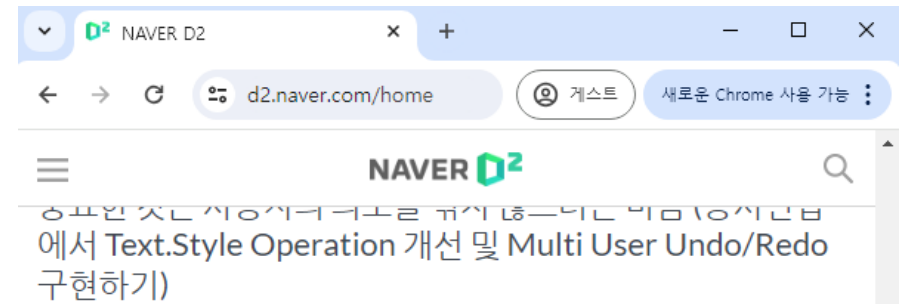
Scala

kafka

TensorFlow

D2의 새로운 소식 받기

구독신청



TOP 5

- 1 일 3,000만 건의 네이버페이 주문 메...
- 2 Virtual Thread의 기본 개념 이해하기
- 3 Golang, 그대들은 어떻게 할 것인가 - ...
- 4 Golang, 그대들은 어떻게 할 것인가 - ...
- 5 백엔드 개발자를 꿈꾸는 학생개발자...

TOP KEYWORDS

빅데이터

분산처리

Golang

Scala

kafka

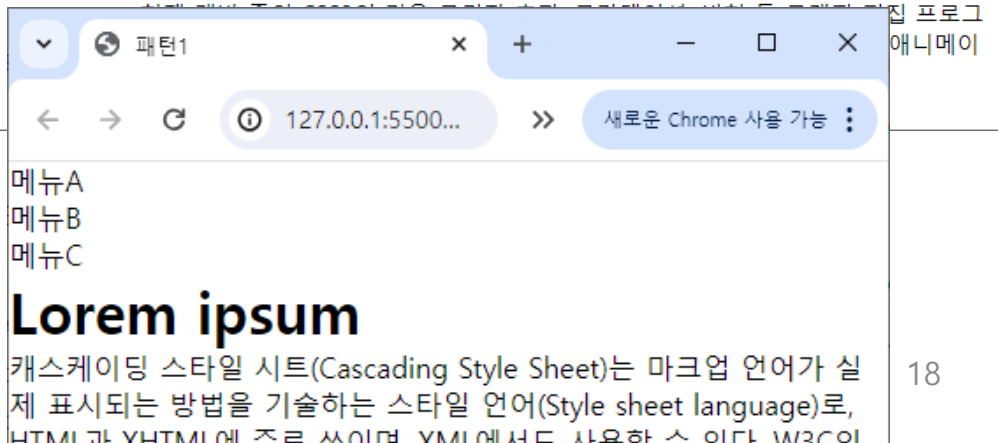
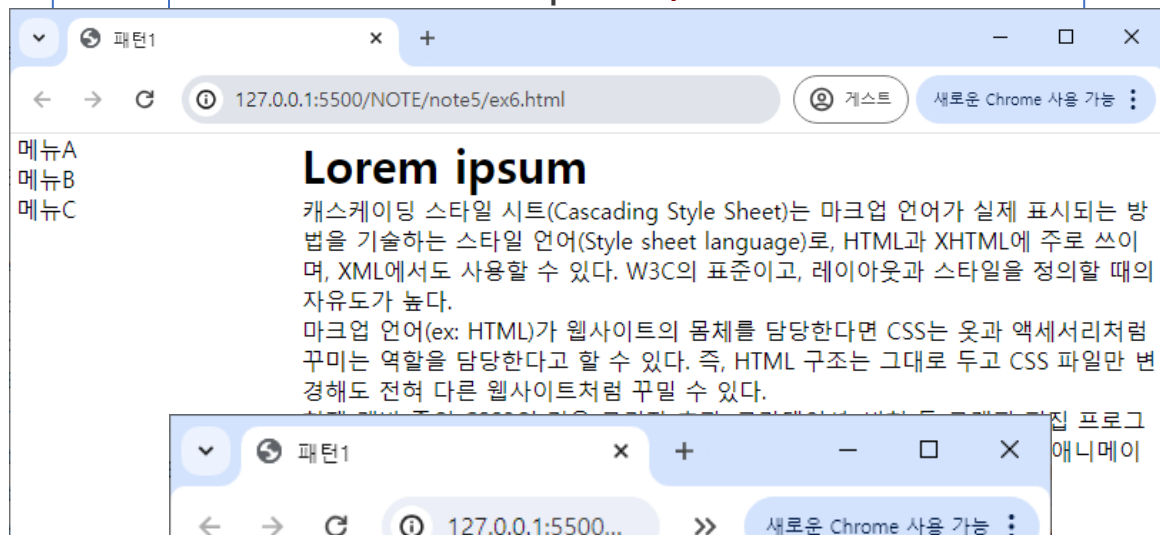
TensorFlow

반응형 웹 패턴

■ HTML 태그 구성은 모바일 장치를 기준으로 작성

```
* { margin: 0; padding: 0; }
body {
  width: 800px;
  margin: 0 auto;
  overflow: hidden;
}
#menu {
  width: 200px;
  float: left;
}
#section {
  width: 600px;
  float: right; /* left로 입력해도 상관없습니다. */
}
li { list-style: none; }
@media screen and (max-width: 767px) {
  /* 스마트폰 사이즈에서는 전부 해제합니다. */
  body { width: auto; }
  #menu { width: auto; float: none; }
  #section { width: auto; float: none; }
}
```

```
<div id="menu">
<ul>
<li>메뉴A</li>
<li>메뉴B</li>
<li>메뉴C</li>
</ul>
</div>
<div id="section">
<h1>Lorem ipsum</h1>
```



그리드 레이아웃

- 반응형 웹 디자인에서 웹 문서 요소를 배치하는 기준
- 웹 사이트 화면을 여러 개의 칼럼(column)으로 나눈 후 웹 요소를 배치
- 화면을 규칙적으로 배열하므로 레이아웃을 일관성 있게 유지할 수 있음

■ 만드는 방법

- 플렉스 박스 레이아웃
 - 수평 방향이나 수직 방향 중 하나를 주축으로 정하고 박스를 배치
 - 여유 공간이 생길 경우 너비나 높이를 적절하게 늘리거나 줄일 수 있음
- css 그리드 레이아웃
 - 수평 방향, 수직 방향 어디로든 배치 가능

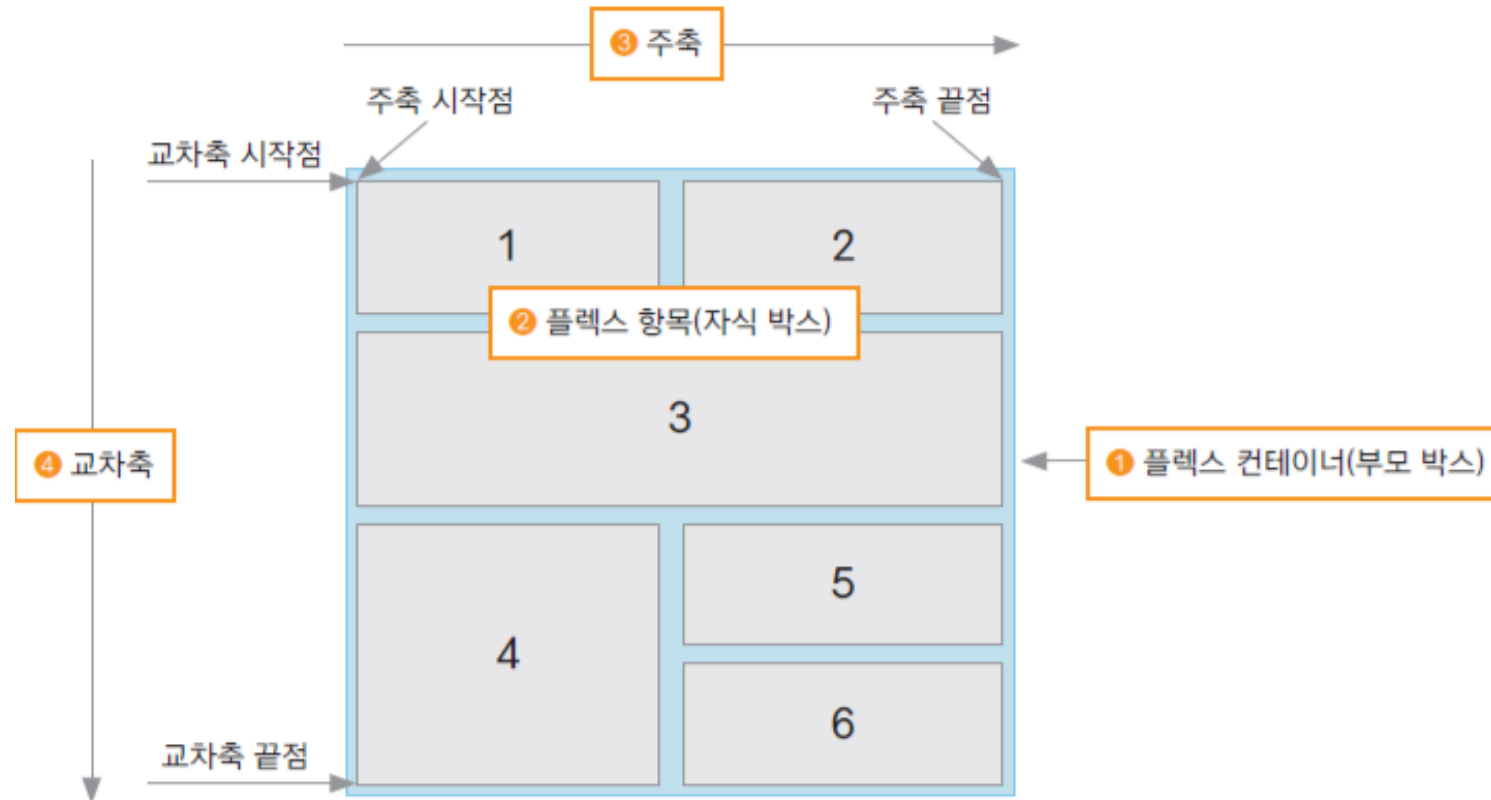
<https://codepen.io/enxaneta/pen/adLPwv>

https://www.w3schools.com/css/css3_flexbox_responsive.asp

플렉스 박스 레이아웃

■ Flex box layout

- 그리드 레이아웃을 기본으로 플렉스 박스를 원하는 위치에 배치
- 여유 공간에 따라 너비나 높이, 위치를 자유롭게 변형 가능



플렉스 박스 레이아웃 주요 속성

■ display 속성

- 값: flex, inline-flex

■ flex-direction 속성

- 플렉스 항목의 배치를 위해 주축과 방향 지정
- 값: row, row-reverse, column, column-reverse

■ flex-wrap

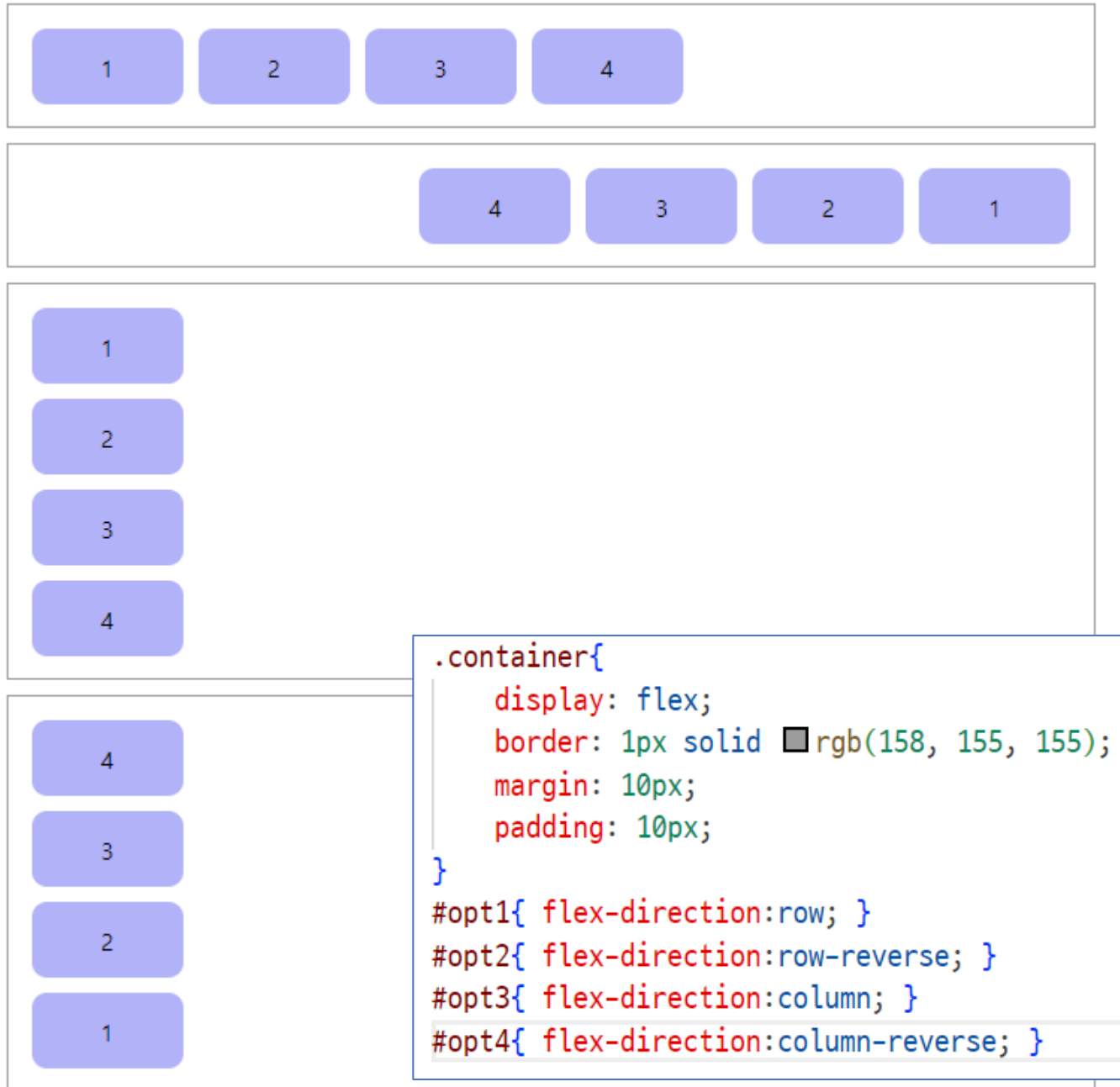
- 플렉스 항목을 한 줄 또는 여러 줄로 배치
- 값: nowrap, wrap, wrap-reverse

■ flex-flow 속성

- 플렉스 배치 방향과 여러 줄 배치를 한꺼번에 지정
- 예: flex-flow: row nowrap

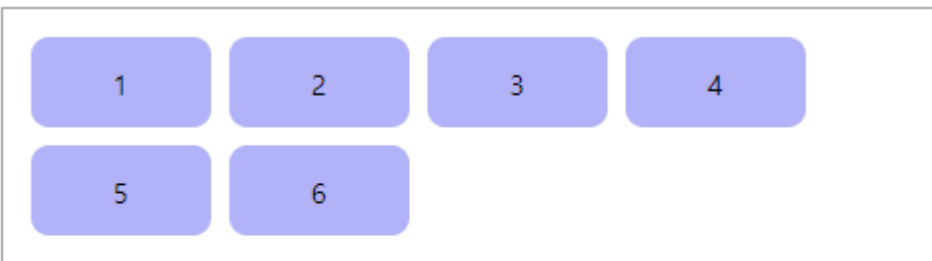
플렉스 박스 예제 #1

```
<div class="container" id="opt1">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
</div>
<div class="container" id="opt2">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
</div>
<div class="container" id="opt3">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
</div>
<div class="container" id="opt4">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
</div>
```



플렉스 박스 예제 #2

```
#opt1{ flex-flow: row wrap; }  
#opt2{ flex-flow: row nowrap; }
```



플렉스 박스 레이아웃 주요 속성

▪ justify-content 속성

- 플렉스 항목을 주축 방향으로 배치할 때의 기준
- 값: flex-start, flex-end, center, space-between, space-around

▪ align-items 속성

- 교차축을 기준으로 배치 방법 조절
- 교차축에서 특정 항목만 지정하고 싶다면 align-self 속성 사용
- 값: flex-start, flex-end, center, baseline, stretch

▪ align-content 속성

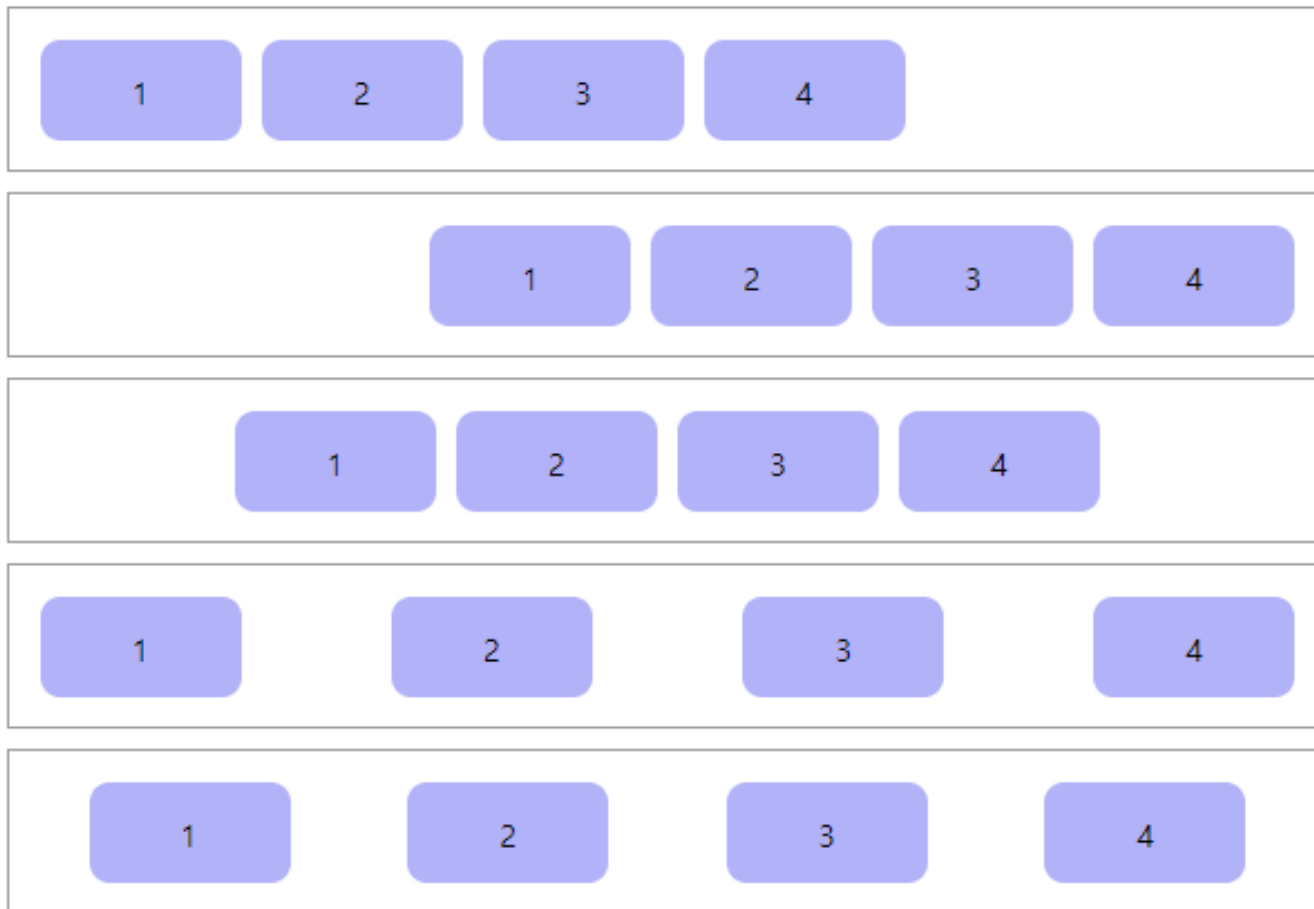
- 플렉스 항목이 여러 줄로 표시될 때 축 기준의 배치 방법 지정
- 값: flex-start, flex-end, center, space-between, space-around, stretch

플렉스 박스 예제 #3

▪ justify-content

- 수평 방향 정렬

```
#opt1{ justify-content: flex-start; }  
#opt2{ justify-content: flex-end; }  
#opt3{ justify-content: center; }  
#opt4{ justify-content: space-between; }  
#opt5{ justify-content: space-around; }
```

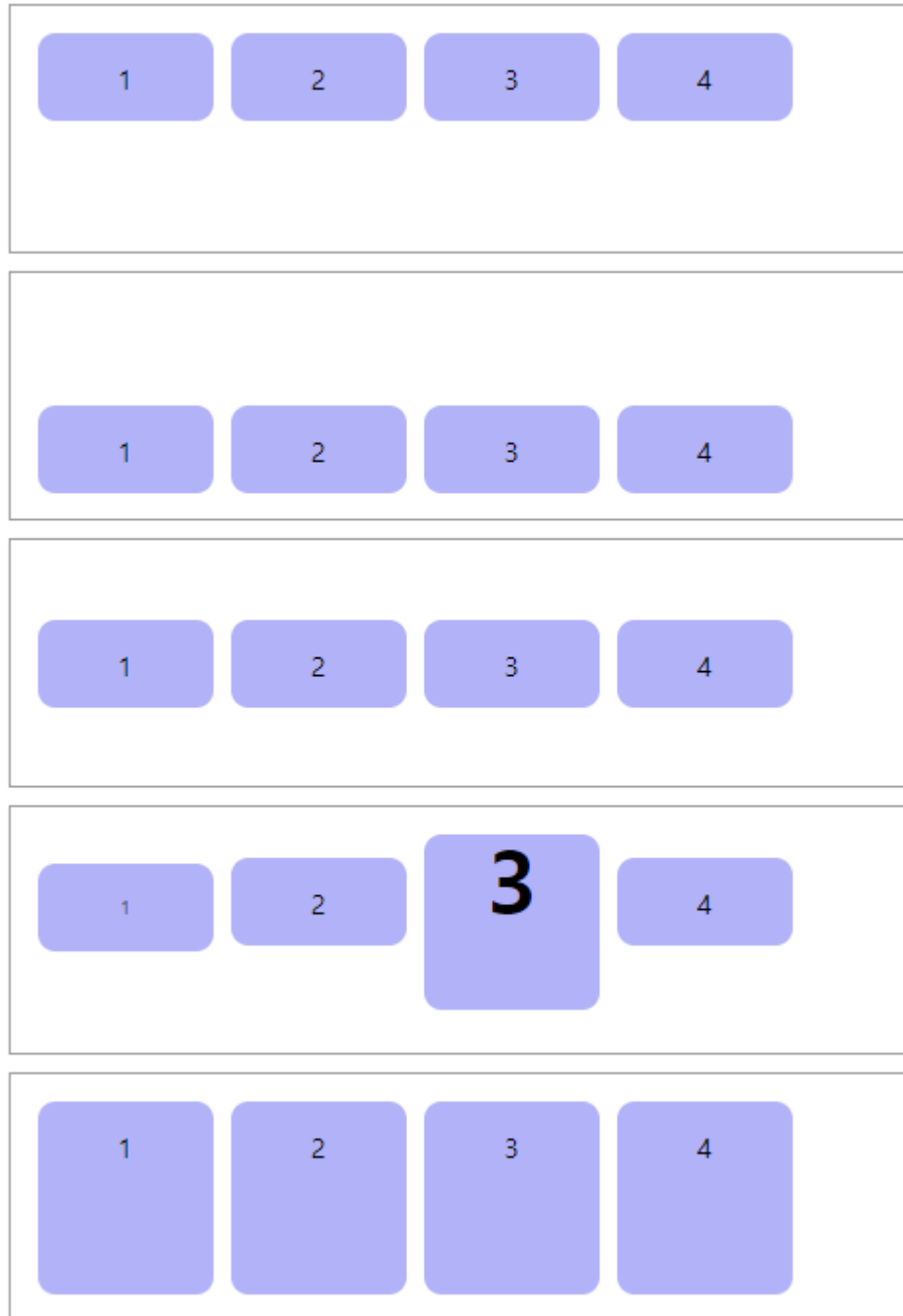


플렉스 박스 예제 #4

■ align-items

- 수직 방향 정렬

```
#opt1{ align-items: flex-start; }  
#opt2{ align-items: flex-end; }  
#opt3{ align-items: center; }  
#opt4{ align-items: baseline; }  
#opt5{ align-items: stretch; }
```



- 플렉스 박스 레이아웃은 주축/교차축 개념이 있지만 CSS 그리드 레이아웃은 양쪽 방향 모두 사용
- 줄(row)과 칼럼(column)으로 화면을 구성하고, 줄 사이의 여백, 칼럼 사이의 여백을 조절.

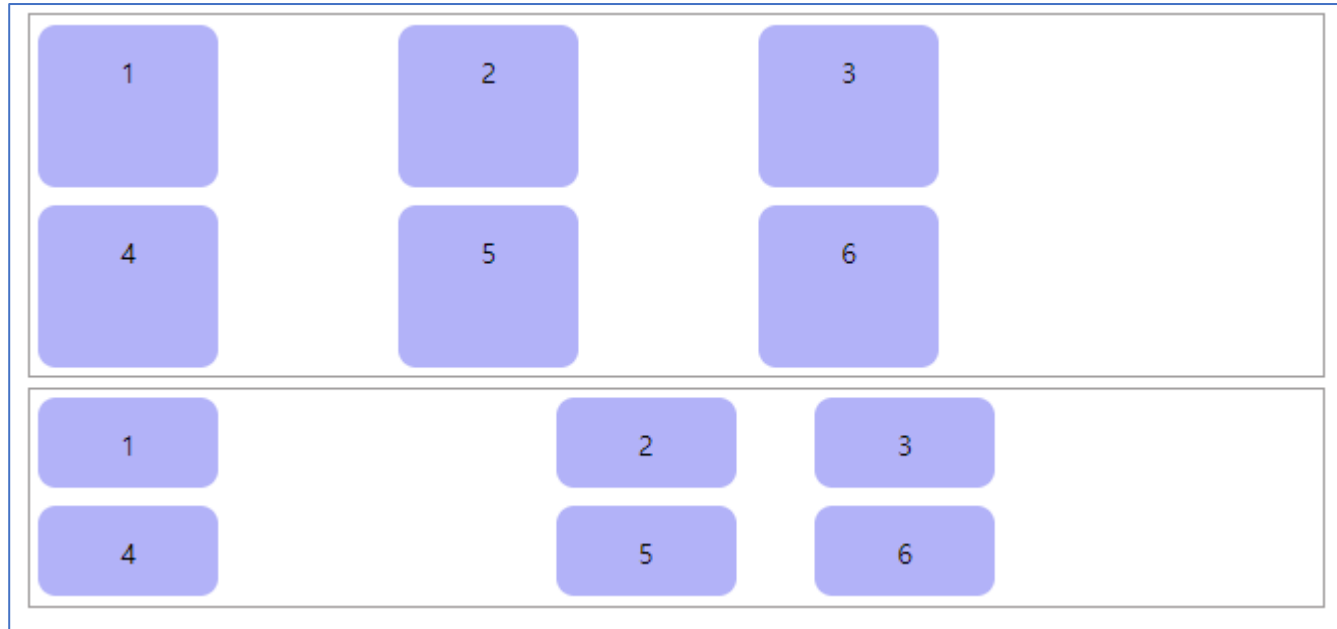


CSS 그리드 레이아웃 항목을 배치하는 속성

- display 속성
 - grid, inline-grid
- grid-template-columns, grid-template-rows 속성

```
#opt1{  
  display:grid;  
  grid-template-columns: 200px 200px 200px;  
  grid-template-rows: 100px 100px;  
}  
  
#opt2{  
  display:grid;  
  grid-template-columns: 2fr 1fr 2fr;  
  grid-template-rows: 1fr 1fr;  
}
```

상대적인 크기를 지정하는 fr 단위



값이 반복된다면 repeat() 함수

grid-template-columns: 1fr 1fr 1fr; → grid-template-columns: repeat(3, 1fr);

CSS 그리드 레이아웃 항목을 배치하는 속성(계속)

■ 자동으로 칼럼 개수를 조절하는 auto-fill, auto-fit

- auto-fit : 남는 공간 없이 꽉 채우기
- auto-fill : 칼럼의 최소 너비까지만 표시하고 남는 공간은 그대로 둠

- 예) 너비 200px인 칼럼을 화면에 가득찰 때까지 배치

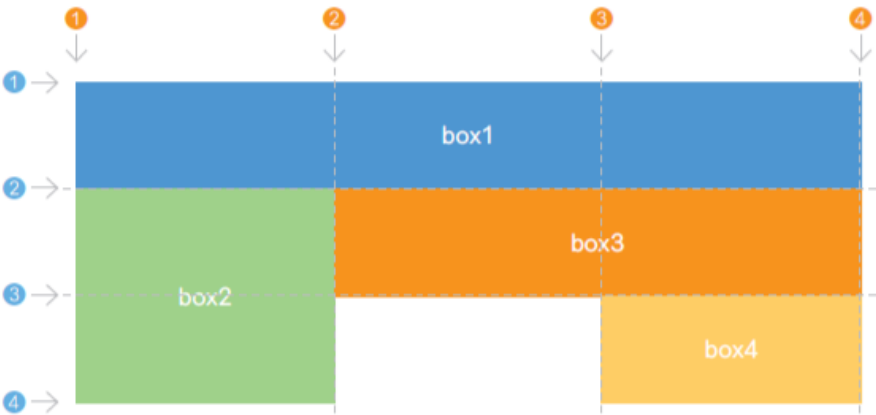
`grid-template-columns: repeat(auto-fit, 200px);`



그리드 라인을 사용한 배치

- CSS 그리드 레이아웃에는 눈에 보이지 않는 그리드 라인이 포함되어 있음
- 그리드 라인을 사용해 그리드 항목을 배치할 수 있음

종류	설명	예시
grid-column-start	칼럼 시작의 라인 번호를 지정합니다.	grid-column-start: 1
grid-column-end	칼럼 마지막의 라인 번호를 지정합니다.	grid-column-end: 1
grid-column	칼럼 시작 번호와 칼럼 끝 번호 사이에 슬래시(/)를 넣어 사용합니다.	grid-column: 1/4
grid-end-start	줄 시작의 라인 번호를 지정합니다.	grid-end-start: 2
grid-row-end	줄 마지막의 라인 번호입니다.	grid-row-end: 4
grid-row	줄 시작 번호와 줄 끝 번호 사이에 슬래시(/)를 넣어 사용합니다.	grid-row: 2/4



```
<style>
.wrapper {
  width: 700px;
  display: grid;
  grid-template-columns: repeat(3, 1fr);
  grid-template-rows: repeat(3, 100px);
}

.box1 {
  background-color: #3689ff;
  grid-column: 1/4;
}

.box2 {
  background-color: #00cf12;
  grid-row: 2/4;
}

.box3 {
  background-color: #ff9019;
  grid-column: 2/4;
  grid-row-start: 2;
}

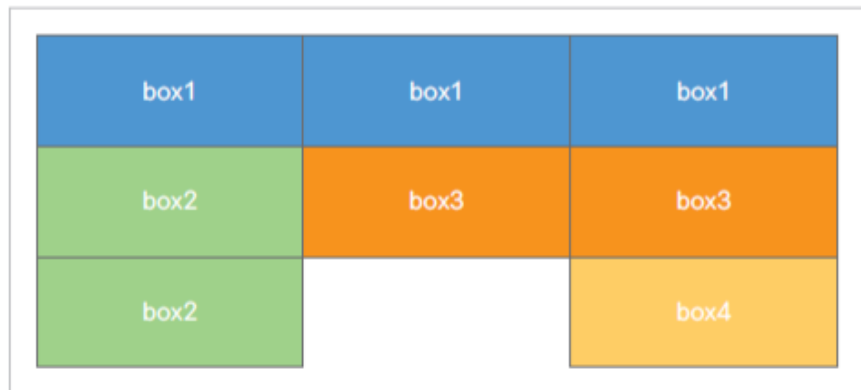
.box4 {
  background-color: #ffd000;
  grid-column-start: 3;
  grid-row-start: 3;
}
```

CSS 그리드 레이아웃

템플릿 영역을 만들어 배치하기

grid-area 속성을 사용해 템플릿 영역을 만든 후 배치

예) box1 ~ box4까지 네 개의 템플릿 영역을 만든 후 배치



1) 템플릿 영역 만들기

```
.box1 {  
  background-color: #3689ff;  
  grid-area: box1;  
}  
.box2 {  
  background-color: #00cf12;  
  grid-area: box2;  
}  
.box3 {  
  background-color: #ff9019;  
  grid-area: box3;  
}  
.box4 {  
  background-color: #ffd000;  
  grid-area: box4;  
}
```

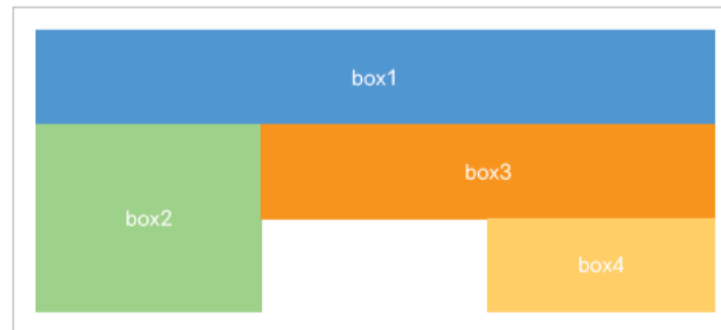
2) 템플릿 영역 배치하기

```
#wrapper {  
  width: 700px;  
  display: grid;  
  grid-template-columns: repeat(3, 1fr);  
  grid-template-rows: repeat(3, 100px);  
  grid-template-areas:  
    "box1 box1 box1"   
    "box2 box3 box3"   
    "box2 . box4";  
}
```

한 줄에 들어갈 템플릿 영역을 큰따옴표(" ")로 묶습니다.

빈 영역은 마침표(.)를 넣습니다.

3) 결과 화면



CSS 그리드 레이아웃으로 갤러리 만들기

사진 제목




사진 설명 : Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ullam, numquam. Ne

사진 제목





사진 설명 : Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ullam, numquam. Neque mollitia esse blanditiis facere.

사진 제목



```
<div id="wrapper">
  <div class="card">
    <header>
      <h3>사진 제목</h3>
    </header>
    <figure>
      
    </figure>
    <p>사진 설명 : Lorem ipsum dolor
    </p>
  </div>

  <div class="card">
```



```
#wrapper{
  display:grid;
  grid-template-columns:repeat(auto-fit, minmax(320px, 1fr));
  grid-gap:1rem;
}
```

```
<div id="wrapper">
  <div class="card">
    <header>
      <h3>사진 제목</h3>
    </header>
    <figure>
      
    </figure>
    <p>사진 설명 : Lorem ipsum dolor
    </p>
  </div>
```

```
<div class="card">
```



사진 제목



사진 설명 : Lorem ipsum dolor sit amet, consectetur adipisicing elit. Ullam, numquam. Neque mollitia esse blanditiis facere.

사진 제목



사진 설명 : Lorem ipsum dolor sit amet, consectetur adipisicing elit. Ullam, numquam. Neque mollitia esse blanditiis facere.

사진 제목



사진 설명 : Lorem ipsum dolor sit amet, consectetur adipisicing elit. Ullam, numquam. Neque mollitia esse blanditiis facere.

사진 제목



사진 설명 : Lorem ipsum dolor sit amet, consectetur adipisicing elit. Ullam, numquam. Neque mollitia esse blanditiis facere.

사진 제목



사진 설명 : Lorem ipsum dolor sit amet, consectetur adipisicing elit. Ullam, numquam. Neque mollitia esse blanditiis facere.

사진 제목



사진 설명 : Lorem ipsum dolor sit amet, consectetur adipisicing elit. Ullam, numquam. Neque mollitia esse blanditiis facere.

사진 제목



사진 설명 : Lorem ipsum dolor sit amet, consectetur adipisicing elit. Ullam, numquam. Neque mollitia esse blanditiis facere.

사진 제목



사진 설명 : Lorem ipsum dolor sit amet, consectetur adipisicing elit. Ullam, numquam. Neque mollitia esse blanditiis facere.