10. 13

코딩 공부법

앞에서 할 때 같이 따라해라 : 따라하고, 시간 줄 때 예제 적용해보고 (이해, 정리, 질문), 응용해보고, 레퍼런스도 찾아봐라.

결국, 내거를 만들어라 (시간 날 때 개인적으로 해보라)

**Float /** Float\_layout.html \* float, fverflow, width 구역 지정 \*

        body{ /\* body 크기 이후 그 안에 section, aside 들어갈 수 있게 수치를 지정해줘야함 \*/

            width: 962px;

            margin: 0 auto;

        }

        #aside{

            width: 200px;

            height: 100%;

            float: right; /\* 오른쪽으로 몰아버리는 float right \*/

        }

        .section {

            width: 760px;

            float: left;

        }

        #wrap { /\* 내용물이 더 넘어오지 못하게 칸 구분

            범위를 넘어오는 아이 윗 태그에 하는게 좋음\*/

            overflow:hidden

        }

    </style>

</head>

<body> <!-- 항상 큰 골격을 먼저, div를 지금은 쓰지만, 나중에는 시멘틱태그를 쓰는게 좋다 -->

            margin: 10px; padding: 10px;

            /\* 태그를 오른쪽으로 붙입니다. \*/

            float: right; /\* float가 오른쪽으로 적용되어 있기 때문에, 1번이 먼저 오른쪽으로 붙게 됨. \*/

        }

Quiz 1, 2, 3 -> 할만함, div 구분 단위를 잘 하자, ., # 아직도 헷갈림 잘 좀 해보자…

**Background : 이미지의 배경화면화**

Bg1.html

    body {

      background-image: url('images/bg1.jpg');

      width: 200px;

      height: 200px;

      /\* 어떤 때에 bg를 쓰는가? 이미지 위에 글자라던가 액션을 넣어야 할 때. \*/

      /\* 그냥 단순한 이미지만 쓰고 싶으면 그냥 img 태그를 사용함  \*/

    }

Bg2.html

    li {

      background-image:url('images/book-icon.png');  /\* 배경 이미지 파일 \*/

      background-repeat:no-repeat;  /\* 배경 이미지 반복 안함 \*/

      background-position:left center;  /\* 배경 이미지 위치 \*/

      padding-left:50px;  /\* 왼쪽 패딩 (박스 모델) \*/

      line-height:40px;  /\* 줄간격 \*/

      /\* bg태그 - 링크 - 이미지 반복 지정 여부 선택 - 이미지 위치 선택 (백그라운드 포지션) -  \*/

    }

Bg3.html

   background-position:right top;  /\* 배경 이미지를 오른쪽 상단에 위치시킴 \*/

    }

    #bg1 {

      background-origin:padding-box;  /\* 패딩까지 배경 이미지 표시 \*/

    }

    #bg2 {

      background-origin:border-box;  /\* 테두리까지 배경 이미지 표시 \*/

    }

    #bg3 {

      background-origin:content-box;  /\* 내용 영역만 배경 이미지 표시 \*/

Bg4.html : bg 사이즈 속성

      background:url('images/bg4.jpg') no-repeat left top;

    #bg1 { background-size:auto;}  /\* 원래 배경 이미지 크기로 표시 \*/

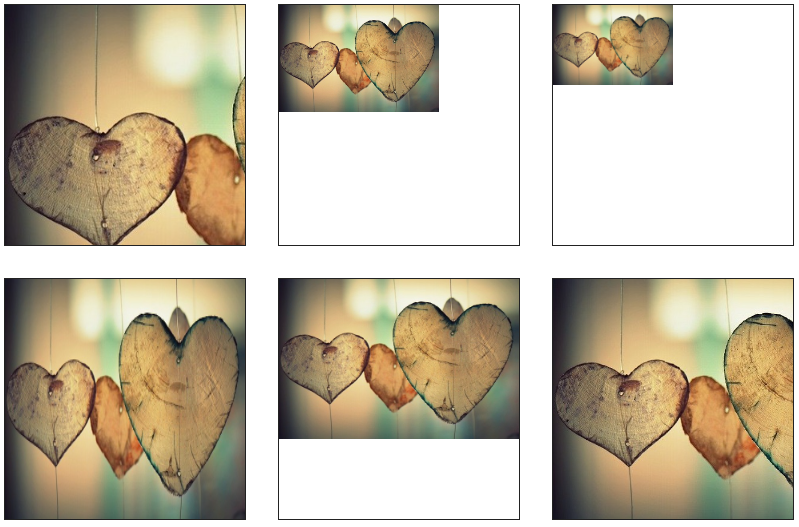
    #bg2 { background-size:200px;}  /\* 너비는 200px, 높이는 자동 계산 \*/

    #bg3 { background-size:50%;}  /\* 배경 이미지 너비는 요소 너비의 50%, 높이는 자동 계산 \*/

    #bg4 { background-size:100% 100%;}  /\* 배경 이미지 너비와 높이는 요소 너비의 100%, 요소 높이의 100% \*/

    #bg5 { background-size:contain;}  /\* 요소 안에 배경 이미지가 다 보이도록 표시 \*/

    #bg6 { background-size:cover;}  /\* 요소를 완전히 덮도록 배경 이미지 표시 \*/



------- css 속성 끝.

**Css 우선순위 ☆**

: **스타일 우선순위** : 인라인 > 내부스타일 > 외부스타일 > 브라우저기본값 순으로 우선순위 적용 됨.

: **선택자 우선순위** : ID선택자 > class 선택자 > 태그 선택자 순으로 나옴

: **!important :** 다 무시하고 가장 최우선으로 적용

: **같은 선택자 가운데 같은 속성 적용시** : 가장 나중에 오는 속성이 적용

: **전체 선택자 > 해당 태그 선택자**

**애니메이션**

**: transform (변형)** : 2차원변형, 3차원 변형으로 이루어짐.

: z축이라는 개념을 알고 있어라.

: translate.html – 2차원 변형 , scale.html – 늘리기, rotate.html – 돌리기

Perspective.html – 3차원, rotate3d.html – 3차원 돌리기

Skew.html – 비틀기

      #movex:hover {

        transform: translateX(50px);  /\* x축으로(가로) 50px 이동 \*/

      }

      #movey:hover {

        transform: translateY(20px);  /\* y축으로(세로) 20px 이동 \*/

      }

      #movexy:hover {

        transform: translate(10px, 20px);  /\* x축으로(가로) 10px, y축으로(세로) 20px 이동 \*/

      }

      #scalex:hover{

        transform: scaleX(20);  /\* x축으로(가로) 2배 확대 \*/

      }

      #scaley:hover{

        transform: scaleY(15);  /\* y축으로(세로) 2배 확대 \*/

      }

      #scale:hover{

        transform: scale(0.1);  /\* x, y축으로(가로, 세로) 0.7배 확대 \*/

      }

      #rotate1:hover {

         transform: rotate(40deg);  /\* 시계 방향(오른쪽)으로 40도 회전 \*/

      }

      #rotate2:hover {

        transform: rotate(-40deg);  /\* 시계 반대 방향(왼쪽)으로 40도 회전 \*/

      }

        .rotatex:hover {

            transform: rotateX(50deg);  /\* x축으로 50도 회전 \*/

        }

        #pers {

            perspective: 300px;  /\* 원근감 추가 \*/

        }

    .origin > div {

      width:100px;

      height:100px;

      background-color:orange;

      transition:all 3s;  /\* 3초 동안 회전하도록 트랜지션 적용 \*/

    }

    #rotatex:hover {

      transform: rotateX(55deg);  /\* x축으로 55도 회전 \*/

    }

    #rotatey:hover {

      transform: rotateY(55deg);  /\* y축으로 55도 회전 \*/

    }

    #rotatez:hover {

      transform: rotateZ(55deg);   /\* z축으로 55도 회전 \*/

    }

    #rotatexyz:hover {

      transform: rotate3d(2.5, 1.2, -1.5, 55deg);  /\* x,y,z축으로 55도 회전, 3d \*/

    }

      #skewx:hover {

        transform: skewX(30deg);  /\* x축 기준으로 30도 비틀기 \*/

      }

      #skewy:hover {

        transform: skewY(15deg);  /\* y축 기준으로 15도 비틀기 \*/

      }

      #skewxy:hover {

        transform: skew(-25deg, -15deg);  /\* x축 기준으로 -25도, y축 기준으로 -15도 비틀기 \*/

      }

**: transition (변화)**

transition-property : 트랜지션의 대상을 지정

transition-duration : 트랜지션을 실행할 시간을 지정

transition-timing-function : 트랜지션의 실행 형태를 지정

transition-delay : 트랜지션의 지연 시간을 지정

transition : transition-property | transition-duration | transition-timing-function | transition-delay 속성을 한꺼번에 지정

tr1.html

      height: 100px;

      background-color: #07f;

      border: 1px solid #222;

      transition-property: width, height, background-color;  /\* 트랜지션 대상 - 너비, 높이 \*/

      transition-duration: 0.5s, 0.5s, 0.5s;  /\* 트랜지션 시간 - 2초 (너비), 1초(높이) \*/

      /\* 무엇을 트랜지션 할 건지, 얼마나 할 건지? \*/

      transition-delay: 2s;

      /\* 몇 초 뒤에 애니메이션을 시작할건지? \*/

      transition-timing-function: linear;

      /\* 동일시간내에 애니메이션이 어떻게 변화되는지에 대한 ex) a->b 로 변하는 색깔 시간의 차이  \*/

속성을 한꺼번에 지정

        background-color: #fb5;

        border: 1px solid #222;

        transition: 1s ease-in;  /\* 대상: all, 시간:2초, 함수:ease-in \*/

: **Annimation**

@keyframes 애니매이션이 바뀌는 지점을 지정 / 미리 선언, 지정을 해놓고 스타일 태그에 넣는 방식

animation-delay 애니매이션의 시작시간을 지정

animation-direction 애니매이션이 종료한 뒤 처음부터 시작할지, 역방향으로 진행할지 지정

animation-duration 애니매이션의 실행시간을 지정

animation-iteration-count 애니매이션 반복 횟수를 지정

animation-name @keyframes로 설정해 놓은 중간 상태를 지정

animation-timing-function 키프레임의 전환 형태를 지정

animation 애니매이션 속성을 한꺼번에 묶어서 지정

        @keyframes shape { /\* shape 애니메이션 정의 \*/

            from {

                border: 1px solid transparent;  /\* 1px짜리 투명한 테두리 \*/

            }

            to {

                border: 1px solid #000;  /\* 검정색 테두리 \*/

                border-radius: 45%;  /\* 테두리를 둥글게 \*/

            }

        }

            animation-name: rotate;  /\* 애니메이션 지정 \*/

            animation-duration: 1s;  /\* 애니메이션 실행 시간 \*/

            animation-iteration-count: infinite; /\* 애니메이션 반복 횟수 \*/

            animation-direction: reverse; /\* 애니메이션 진행 방향 \*/

    @keyframes rotate {  /\* 0도 -> x축 -180도 회전 -> y축 -180도 회전 \*/

      from { transform: perspective(120px) rotateX(0deg) rotateY(0deg); }

      50% { transform: perspective(120px) rotateX(-180deg) rotateY(0deg); }

      to { transform: perspective(120px) rotateX(-180deg) rotateY(-180deg); }

    }

\*transform이 꼭 필수는 아님! 안해도 됨!\*

register.html \*인라인-블록 중요함\*

    display: block;

    margin: 0 auto; /\* 가운데 정렬하는 애인데 인라인태그에는 맞지가 않음 \*/

그래서 display로 block로 바꿔줌

**flex**

주축(가로), 교차축(세로) / justify-content, align-items

Flex-1.html \*flex item 정렬\*

      display: flex; /\* 안에 있는 item(box) default가 수평으로 정렬 됨 \*/

      /\* display: grid; \*/

    }

    .box {

      padding: 5px 45px;

      margin: 5px;

      width: 80px;

      background-color: #222;

    }

    p {

      color: #fff;

      text-align: center;

    }

    /\* flex direction : 방향 지정 \*/

    #opt1 {

      flex-direction: row; /\* dafault, 왼 -> 오 \*/

    }

    #opt2 {

      flex-direction: row-reverse; /\* 오->왼 \*/

    }

    #opt3 {

      flex-direction: column; /\* 위 -> 아래 \*/

    }

    #opt4 {

      flex-direction: column-reverse; /\* 아래 -> 위 \*/

    }

Flex-2.html **flex-wrap** \*flex 줄바꿈\*

    #opt1{ /\* flex-wrap:아이템들이 넘칠 때 감싸는 기능 \*/

      flex-wrap: nowrap; /\*default \*/

      /\* 칸이 줄어도 그래도 한줄에 있음, 애들 사이즈가 작아짐 \*/

    }

    #opt2{

      flex-wrap: wrap;

      /\* 밑으로 넘김 \*/

    }

    #opt3{

      flex-wrap: wrap-reverse;

      /\* 밑으로 넘기는거를 반대로 감싼다 \*/

    }

Flex-3.html **flex-flow**

    #opt1{ /\* flex-flow: wrap, direction 한번에 가능  \*/

      flex-flow: row wrap;   /\* 왼쪽에서 오른쪽, 여러 줄 \*/

    }

    #opt2{

      flex-flow: row nowrap;  /\* 왼쪽에서 오른쪽, 한 줄 \*/

    }

Flex4.html **justify-content** \*flex 주축 기준 배치

    #opt1{ /\* justify-content : 주측 (가로측)을 기준으로 배치하는 아이 \*/

      justify-content: flex-start  /\* 주축 시작점 기준으로 배치 \*/

    }

    #opt2{

      justify-content: flex-end   /\* 주축 끝점 기준으로 배치 \*/

    }

    #opt3{

      justify-content: center     /\* 주축 중앙 기준으로 배치 \*/

    }

    #opt4{

      justify-content: space-between /\* 시작점과 끝점 배치 후 중간 항목은 같은 간격으로 배치 \*/

    }

    #opt5{

      justify-content: space-around  /\* 전체 항목을 같은 간격으로 배치 \*/

    }

Flex5.html **align-items** \*flex 교차축 기준 배치

    #opt1{ /\* align-item : cross (세로)축을 기준으로 배치 \*/

      align-items: flex-start;  /\* 교차축 시작점 기준으로 배치 \*/

    }

    #opt2{

      align-items: flex-end ;    /\* 교차축 끝점 기준으로 배치 \*/

    }

    #opt3{

      align-items: center ;   /\* 교차축 중앙 기준으로 배치 \*/

    }

    #opt4{

      align-items: baseline;    /\*  문자 기준선에 맞춰 배치 (폰트 사이즈 위 아래 그만큼 추가) \*/

    }

    #opt5{

      align-items: stretch;      /\* 항목을 늘려 교차축에 가득차게 배치 \*/

    }

Flex6.html **align-self** \*교차축 정렬을 다르게

        display:flex;         /\* 플렉스 컨테이너 지정 \*/

        align-items: center;  /\* 교차축의 중앙에 배치 \*/

      }

      .box {

        padding:5px 45px;

        margin:5px;

        background-color:#222;

      }

      #box1 {

        align-self: flex-start;  /\* 교차축의 시작점에 배치 \*/

      }

      #box3 {

        align-self:stretch;       /\* 교차축에 가득 차게 늘림 \*/

      }

Flex7.html **align-content \*교차축**

      display: flex;          /\* 플렉스 컨테이너 지정 \*/

      flex-flow: row wrap;   /\* 왼쪽에서 오른쪽, 여러 줄 표시 \*/

      border: 1px solid #222;

      background-color: #eee;

      margin: 30px;

    }

    /\* align-content 주축에서 줄바꿈이 생겼을 때 플렉스 항목을 여러줄로 표시할 때 사용  \*/

    /\* 덩어리진 하나의 컨텐츠들을 한번에 바꿀때 필요함 \*/

**center.html**

      display: flex; /\* flex 선언 \*/

      justify-content: center; /\* 중심축 중간 설정 \*/

      align-items: center; /\* align 중간 설정 \*/

      min-height: 100vh; /\* 최소 높이 설정 \*/

VW, VH 뷰 포트의 width, height (ex. 10vw = 현재 뷰 포트에서 10%)

MAX, MIN 속성

- max-width, max-height

최대 너비(높이), 콘텐츠의 너비(높이)가 max-width보다 더이상 커지지 않음

- min-width, min-height

최소 너비(높이), 콘텐츠의 너비(높이)가 min-width보다 작아지면 더 이상 줄어들지 않음

[flex-grow](https://www.w3schools.com/cssref/css3_pr_flex-grow.asp) / [flex-shrink](https://www.w3schools.com/cssref/css3_pr_flex-shrink.asp) / [flex-basis](https://www.w3schools.com/cssref/css3_pr_flex-basis.asp) / flex: *flex-grow* *flex-shrink* *flex-basis*

     /\*

   flex : 너비설정

   flex-basis : 기본크기

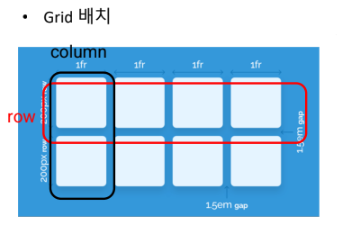
   flex-grow : 컨테이너에 남는 공간이 있으면, item의 크기가 얼마나 더 할당 가능한지 나타내는 속성

  flex-shrink : 컨테이너에 공간이 부족해지면 item의 크기가 얼마나 더 줄어들 수 있는지 나타내는 속성

      \*/

**grid**

column, row 개념

 가로 개념 : 1fr

Grid-1.html \*기본 개념

      display: grid;

      /\* grid-template-columns: 200px 200px 200px 200px; 컬럼 (열)에 대한 부분 지정, 쓴 갯수만큼 열이 생김 \*/

      /\* grid-template-columns: 1fr 1fr 1fr; (비율-전체를 3개로 나눔)  \*/

      grid-template-columns: repeat(5, 1fr); /\* 1fr만큼 크기를 3번 반복 \*/

      grid-template-rows: 200px 200px; /\* 로우(행)에 대한 부분 지정 \*/

      grid-template-rows: minmax(100px, auto) /\* 100px 미만으로 안작아짐. 글자 크기 따라 알아서 커짐 \*/

  }

Grid-4.html \*그리드 간격

      grid-gap: 20px 30px;  /\* 칼럼 간격 30px, 줄 간격 20px  \*/

      /\* 또는 아래와 같이 사용 \*/

      /\* grid-column-gap:30px; \*/

      /\* grid-row-gap:20px; \*/

Grid-6.html \*9칸 실습 – areas로 만들어서 제작함

    grid-template-rows: repeat(4, 100px); /\* 컬럼 3개, 로우 3개 \*/

    grid-template-areas:

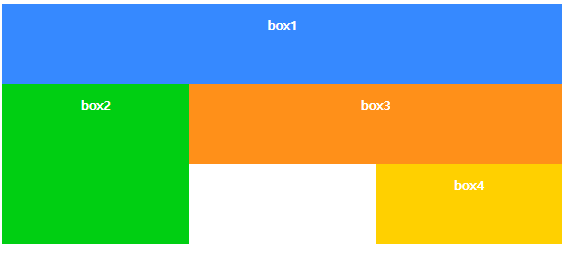
    "box2 box4 box4 box3"

    "box2 box4 box4 box3"

    "box2 box4 box4 box3"

    "box2 box4 box4 box3" /\* 비어있는 공간은 .찍기 \*/

    /\* area를 사용하는 방법은, 테트리스 처럼 묶여있어야함 'ㅁ' '----' '|'모양이 되어야함 \*/



box이름 grid 걸어두고, grid template, areas로 크기를 지정해놓고 그리드를 만들어냄.

Grid-5.html \*칸 만들기 실습 – 숫자로 만들어서

 .box1 {

    background-color: #3689ff;

    grid-column: 1/4; /\* 실습표에 나와있는 1~4까지 영역을 차지하겠다! \*/

  }

  .box2 {

    background-color: #00cf12;

    grid-row: 2/4; /\* row로 2~4까지 영역을 차지하겠다! \*/

  }

  .box3 {

    background-color: #ff9019;

    grid-column: 2/4;

    grid-row-start: 2; /\* 그리드의 시작 위치를 정해줌 (2행의 2번째부터 시작하기 때문에) \*/

    }

  .box4 {

    background-color: #ffd000;

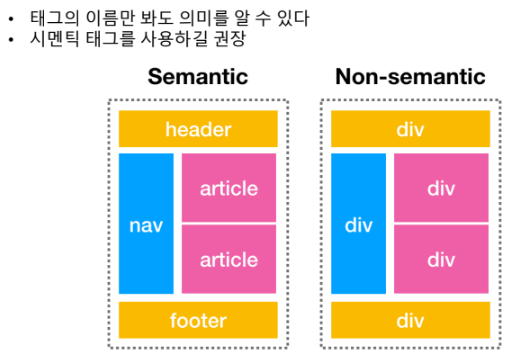
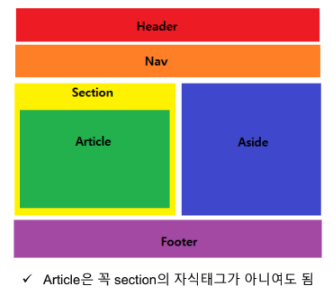
    grid-row-start: 3;

    grid-column-start: 3; /\* 3행, 3열에서 시작하는 box! \*/

  }

**시멘틱태그**

<Div>가 너무 많으면 헤더인지, 메인인지 알기가 복잡하다. 점점 시멘틱 태그를 쓰는 것을 권장함.

Nav : 메뉴

Float\_latout.html – 시멘틱 태그로 바꾸기 semantic\_layout.html 참고

**내일 큰거온다**