12강 반응형 레이아웃 패턴

1) 해상도별 미디어 쿼리

① 일반 가로형
@media screen and (orientation: landscape) { }

② 일반 세로형
@media screen and (orientation: portrait) { }

③ 스마트폰 (가로/세로)
@media only screen and (min-device-width: 320px) and (max-device-width: 480px) { }

④ 스마트폰 (가로)
@media only screen and (min-width: 321px) { }

⑤ 스마트폰 (세로)
@media only screen and (max-width: 320px) { }

⑥ iPad (가로/세로)
@media only screen and (min-device-width: 768px) and (max-device-width: 1024px) { }

⑦ iPad (가로)
@media only screen and (min-device-width: 768px) and (max-device-width: 1024px) and (orientation: landscape) { }

⑧ iPad (세로)
@media only screen and (min-device-width: 768px) and (max-device-width: 1024px) and (orientation: portrait) { }

2) 가장 많이 사용하는 유동형 패턴 (Mostly Fluid)

Mostly Fluid 패턴은 반응형 웹디자인에서 가장 많이 사용하는 패턴입니다.

- Wide PC에서는 가변이 아닌 고정형으로 가고 이보다 작아지면 내부는 가변적으로 변하게끔 설정하는 패턴입니다.
- 태블릿 PC에서는 여백을 없애고 가로가 100%인 상태로 콘텐츠를 노출합니다.
- 모바일에서는 작은 크기에서 최적화하기 위해 column들을 한 개씩 세로로 정렬 (vertical align) 합니다.

1201.html :

```
<style>
* {margin:0; padding:0;}
#wrap {margin:0 auto; width:1000px;}
#box_a {height:400px; background:#f1779e;}
#box_b {float:left; width:50%; height:400px; background:#fcb354;}
#box_c {float:right; width:50%; height:400px; background:#24bbaf;}
#box_c p {float:left; margin:1.0em;}

@media screen and (max-width: 1024px) {
    #wrap {width:90%;}
}
@media screen and (max-width: 800px) {
    #wrap {width:100%;}
}
@media screen and (max-width: 480px) {
    #box_b {float:none; width:auto;}
    #box_c {float:none; width:auto;}
}
</style>
```

3) 컬럼 드롭 패턴 (Column Drop)

Column Drop 패턴에서는 가로로 정렬 (Inline align)되어 있던 요소가 해상도에 따라 세로로 정렬되는 모습을 볼 수 있습니다.

1202.html :

```
<style>
* {margin:0; padding:0;}
#box_a {float:left; width:30%; height:400px; background:#fcb354;}
#box_b {float:left; width:50%; height:400px; background:#f1779e;}
#box_c {float:right; width:20%; height:400px;  background:#24bbaf;}

@media screen and (max-width: 1024px) {
    #box_b {float:none; width:auto;}
    #box_c {float:none; clear:both; width:auto;}
}
@media screen and (max-width: 800px) {
    #box_a {float:none; width:auto;}
}
</style>
```

4) 시프터 패턴 (Shifter)

세로와 가로 정렬이 혼합된 모습으로 PC 웹 사이트와 모바일 웹 사이트 사이의 GNB 영역 (Global Navigation Bar : 페이지마다 항상 위치하게 되는 주메뉴 영역)의 변화가 가장 큽니다. 자칫하면 다른 사이트로 보일 수도 있으나, 이질감이 느끼지 않게 룩엔픽 (Look & Feel)을 유지하면서 변형한다면 오히려 사용자들에게 참신하고 좋은 느낌을 줄 수 있습니다.

Shifter 패턴은 Mostyle Fluid 패턴처럼 큰 화면에서는 고정 픽셀이고 작은 화면에서는 퍼센트 형으로 변경되는 특징이 있습니다.
#box_a 요소는 GNB 역할로 왼쪽에 있다가 모바일 버전으로 가면서 위로 자리를 변경하게 됩니다.
레이아웃을 크게 변경하면 내부에 들어가는 요소의 디자인도 많이 변경해야 하므로 번거롭고, 새롭게 디자인해야 한다는 단점이 있습니다.

1203.html :

```
<style>
* {margin:0; padding:0;}
#wrap {margin:0 auto; width:1000px;}
#box_a {float:left; width:20%; height:400px; background:#fcb354;}
#box_b {float:left; width:80%; height:200px; background:#f1779e;}
#box_c {float:left; width:80%; height:200px;  background:#24bbaf;}

@media screen and (max-width: 1024px) {
    #wrap {width:90%;}
}
@media screen and (max-width: 800px) {
    #wrap {width:100%;}
    #box_a {float:none; width:100%;}
    #box_b {float:none; width:100%;}
    #box_c {float:none; width:100%;}
}
</style>
```

5) 작은 리스트 패턴 (Tiny Tweaks)

PC부터 모바일까지 모두 같은 형태로 디자인하는 방식으로, 간단한 사이트 또는 리스트 형태로 꾸며진 사이트에 사용할 패턴으로 적합니다.

Tiny 패턴은 PC 버전부터 모바일까지 같은 형태로 작아지기만 하는 레이아웃이므로 자칫하면 꽤 지루하게 보일 가능성이 높습니다. 따라서 내부에 있는 리스트를 변경함으로써 지루함을 없애는 게 이 패턴의 노하우입니다. 예시로 만든 코드 역시 내부의 리스트가 가로 5개에서 4개, 3개, 2개로 점차 줄어들게 했습니다.

우선 ul, li 요소로 리스트를 구성했으면 ul 요소는 width 값을 80%로 설정하고 가운데 정렬을 했습니다.
지금부터가 중요한 부분입니다. 우서 내부에 들어가는 li 요소를 가로로 배치하기 위해 float 속성을 left로 설정했습니다. 이렇게 하면 한 줄에 몇 개의 li 요소가 노출되는가는 li 요소의 width 값에 따라 달라집니다. li 요소의 width 속성의 합이 ul 요소의 width 값보다 커지면 자동으로 줄바꿈되어서 나열되기 때문입니다.

정확히 4개만 노출되기를 바란다면 margin 값까지 고려해서 계산해야 합니다.

① 5개 노출되는 wide PC의 경우
(100%-10%)/5=18%;
100% : li 요소를 감싼 ul 요소의 너비
10% : 각 li 요소의 margin 값의 합
5 : li 요소의 개수

② max-width:1024
(100%-8%)/4=23%;

③ max-width:800px
(100%-6%)/3=31.333 ...%;

④ max-width:480px
(100%-4%)/2=48%;

이런 식으로 반응형에 따른 리스트 개수를 조절하는 방법을 활용하면 갤러리 리스트 또는 네비게이션 메뉴에서 활용할 수 있습니다.

1204.html :
```
<style>
* {margin:0; padding:0;}
body {font-family:Arial, sans-serif;}
ul {list-style:none;}
#box_a {height:400px; background:#f1779e;}
```

```
.list_man {margin:0 auto; padding-top:50px; width:80%;}
.list_man li {float:left; margin:1%; width:18%; background:#fff;}

@media screen and (max-width: 1024px) { /* li 4개 */
    .list_man {width:95%;}
    .list_man li {margin:1%; width:23%;}
}
@media screen and (max-width: 800px) { /* li 3개 */
    .list_man li {margin:1%; width:31.333%;}
}
@media screen and (max-width: 480px) { /* li 2개 */
    .list_man li {margin:1%; width:48%;}
}
</style>
```

5) 오프 캔버스 패턴 (Off Canvas)

PC 웹 사이트를 모바일 디바이스로 가져오면서 콘텐츠의 크기와 양을 줄이지 않는 새로운 방법으로, 일부 영역을 숨겨서 사용자의 제스쳐 혹은 액션에 따라 숨겨뒀던 영역을 보여주는 방식입니다. 작은 디바이스에서 많은 것을 보여주는 방법보다 더 편한 UX를 적용해 사용하며 페이스북 모바일 웹, 앱 등에서 많이 사용하고 있는 패턴입니다.

오프 캔버스 패턴은 단순하게 HTML과 CSS로 이뤄지는 경우는 없습니다. 대부분 자바스크립트와 제이쿼리 등을 이용해 작업합니다.

max-width 800px 영역에서 각 부분의 width 값의 합은 절대로 100%가 될 수가 없으며 이는 UI 개발을 하는 이에게 피로를 주는 주범 중 하나입니다. 그러므로 간단하게는 세 개 중 한 개를 33.334%로 설정해 해결할 수 있습니다.

1205.html :

```
<style>
* {margin:0; padding:0;}
html, body {overflow-x:hidden; width:100%;}
#wrap {margin:0 auto; width:1000px;}
#wrap_box {display:block; width:100%;}
#box_a {float:left; width:30%; height:400px; background:#fcb354;}
#box_b {float:left; width:40%; height:400px; background:#f1779e;}
#box_c {float:left; width:30%; height:400px; background:#24bbaf;}

@media screen and (max-width: 1024px) {
    #wrap {width:100%;}
}
@media screen and (max-width: 800px) {
    #wrap_box {position:absolute; width:150%;}
    #box_a {width:33.333%;}
    #box_b {width:33.334%;}
    #box_c {width:33.333%;}
}
@media screen and (max-width: 480px) {
    #wrap_box {margin-left:-100%; width:300%;}
}
</style>
```