11강 반응형 웹 코딩의 기초

우리는 PC 웹 사이트, 태블릿 PC 그리고 스마트폰 등 다양한 디바이스에서 웹 서핑을 즐기고 있습니다. 다양한 해상도에 맞는 화면을 보고 실무자라면 그냥 지나쳐서는 안 됩니다.
반응형 웹디자인은 단순하게 기획자만 알아야 할 사항이 아닌, 디자이너, UI 개발자, 개발자까지 전부 알고 있어야만 서로 간의 의사소통을 문제없이 진행할 수 있고 프로젝트도 성공적으로 진행할 수 있습니다.

1) IA (Information Architecture) 설계

먼저 페이지에 들어갈 요소를 정확히 설계해야 합니다. 다양한 정보 요소가 들어가는 것은 문제가 없지만, 해상도에 따라 요소를 변경해야 할 때가 있습니다. PC 웹 사이트에서는 많은 정보를 담을 수 있지만, 모바일 디바이스에서는 작은 화면에 너무 많은 내용을 넣게 되면 불편하기 때문입니다.
따라서 PC 웹 사이트와 모바일 웹 사이트의 정보설계를 할 때는 많은 고민을 하고, 정확한 타깃을 정해서 설계하는 것이 무엇보다 중요합니다.

멀티미디어 콘텐츠 역시 PC와 모바일에서 처리하는 방법이 다르므로 어떤 식으로 정보를 보여줄 것인지 확인이 필요합니다.

2) UI (User Interface) 설계

해상도의 변화에 따라 레이아웃 형태를 변형하고, 각 요소의 위치도 변경하게 되므로 반응형 웹디자인에 맞출 수 있게 정확한 가변 해상도를 정하고 설계에 들어가야 합니다.
기본적으로 PC 웹 사이트, 태블릿 PC 웹 사이트 그리고 스마트폰 웹 사이트 3단계 해상도는 반드시 고려해야 합니다. 더 다양한 디바이스에 최적화하려면 앞서 말한 기본 3단계에서 좀 더 세분화하는 방법도 있습니다.

이 외에도 모바일향 반응형 웹디자인을 지향하는 방법도 있습니다. 기존 PC 웹 사이트는 제외하고 모바일 디바이스 (태블릿 PC, 스마트폰) 웹 사이트만 따로 반응형 웹디자인으로 제작하는 방법입니다.
예를 들어, 네이버는 www.naver.com과 m.naver.com 두 가지 버전의 사이트를 운영하고 있으며 m.naver.com은 반응형 웹디자인을 이용해 태블릿 PC와 스마트폰에 대응하고 있습니다. 대응하는 해상도는 총 3가지로 iPad ladnscape, iPad Portrait, iPhone 이렇게 3가지로 생각하는 것이 이해하기 쉽습니다.
네이버는 대형 포탈 사이트이므로 PC 웹 사이트에서는 많은 정보를 담아야 합니다. 하지만 모바일 디바이스에서는 많은 양의 콘텐츠를 담게 되면 여러 가지 문제가 발생하므로 따로 제작하는 것이 좋습니다. 물론 반응형 웹디자인을 적용해서 PC 웹 사이트의 콘텐츠 중 일부를 모바일 디바이스에서 사라지게 할 수도 있으나 이 방법은 소스 낭비가 심하며, 모바일 디바이스 사용자에게는 필요 없는 HTML 소스까지 가져오게 하는 문제도 있습니다.

3) UI (User Interface) 디자인기획

PC 웹 사이트는 이미지 크기와 글자 크기 등 나름 정해진 규칙 속에서 제작해 왔습니다. 하지만 모바일 디바이스는 성장한 지 얼마 되지 않았기에 이미지 크기, 글자 크기 등을 나름대로 고민해서 작성해야 합니다.

이미지들은 Flexible Images 기법을 이용해 가변적으로 작업할지, 단순 픽셀로 나타나게 할지 확실하게 결정해야 합니다. 물론 해상도에 따라 유연 (flexible)하게 할지 고정 형태로 갈지를 변경할 수도 있습니다.

네비게이션 역시 고정형 스타일로 만들지, 아니면 퍼센트를 활용해 해상도마다 다르게 노출할 것인지 고려해야 합니다.

그리고 모바일 디바이스에서 가장 중요한 점 중 하나는 터치 영역 부분과 터치를 활용한 인터렉션 부분입니다. 기존 PC 웹 사이트 역시 작은 버튼을 클릭하기 쉽게 일부러 클릭 영역을 버튼보다 조금 더 크게 설정하곤 했습니다. 터치도 최소 크기를 정하고, 최소 크기보다 작은 영역일 때는 UI 개발로 해결해 나가는 것도 UI 디자인 기획 당시부터 고려해야 합니다. 반응형 웹디자인이 널리 퍼지면서 많은 사이트가 반응형 웹디자인을 적용했고, 자연스럽게 패턴이 생기게 되었습니다.

반응형 웹디자인 패턴으로는 Mostly Fluid, Column drop, Shiter, Tiny, Off canvas 등이 있습니다.

여기에 나온 패턴 모양대로 전부 진행되는 것은 아니지만, 반응형 웹디자인을 적용한 모든 사이트가 이를 토대로 움직이고 있습니다. 이에 해당하지 않은 패턴이라고 반응형 웹디자인으로 불리지 않는 것은 아니지만 흐름의 어색함 때문에 사용자가 불편함을 느낄 수도 있습니다.

5가지 패턴을 유심히 보면 5가지 패턴 모두 모바일 버전에서는 column이 세로로 정렬 (vertical align)된 모습을 볼 수 있으며 오른쪽 아래 요소가 점차 아래로 내려가는 모습도 보일 것입니다.

이렇게 반응형 웹디자인이 변형되는 과정은 사실 자연스러운 부분입니다. 각 요소를 마크업 (Mark-up)하고 각 요소가 노출되는 순서는 브라우저에서 왼쪽 위부터 시작되기 때문입니다. 왼쪽 위 영역부터 오른쪽 아래로 차례대로 나열되므로 가로 폭이 작아져서 축소된다고 하면 자연스럽게 오른쪽 아래의 자리가 없어서 아래로 내려가는 모습을 상상할 수 있을 것입니다.

1101.html :

```
<style>
body {
	margin: 20px;
	padding: 20px;
	line-height: 1;
	font-family: "Open Sans", sans-serif;
	font-size: 1em;
	background: #555;
	color: #000;
}
h1.title {
	margin: 0;
	padding: 0;
	font-size: 1.5em;
	font-weight: 300;
}
h2 {
	display: none;
	margin-top: 33px;
	font-size: 16px;
	font-weight: normal;
	color: #fff;
}

@media screen and (orientation: landscape) {
	body {
		background: #8ac007;
	}
	#landscape {
		display: block;
	}
}
@media screen and (orientation: portrait) {
	body {
		background: #24bbaf;
	}
	#portrait {
		display:  block;
	}
}
</style>
```

1102.html :

```
<style>
body {
	margin: 20px;
	padding: 20px;
	line-height: 1;
	font-family: "Open Sans", sans-serif;
	font-size: 1em;
	background: #555;
	color: #000;
}
.title {
	margin: 0;
	padding: 0;
	font-size: 1.5em;
	font-weight: 300;
}
#wrapper {
	margin-top: 30px;
	color: #8ac007;
}

/* 기본 media query */
@media screen and (orientation: landscape) {
	#wrapper:after {
		content: "orientation: landscape";
		display: block;
	}
}
@media screen and (orientation: portrait) {
	#wrapper:after {
		content: "orientation: portrait";
		display: block;
	}
}
/*
@media screen and (max-width: 400px) {
	#wrapper:after {
		content: "max-width: 400px";
		display: block;
	}
}
```

```
*/
</style>
```