NEXCORE

Alopex UI 2.3

Alopex UI 퍼블리싱가이드

Version 1.0.1

SK 주식회사 C&C

제•개정 이력

제•개정 번호	제•개정 이력	제•개정 일자
1.0.0	최초 작성	2015-04-01
1.0.1	현행화	2016-01-27

목 차

©Copyright 2016 SK 주식회사 C&C. All rights reserved.

©Copyright 2016 SK 주식회사 C&C. All rights reserved.

1. 문서 개요

1.1. 목적

본 문서는 프로젝트에서 Alopex UI 프레임워크 (이하 Alopex) 를 활용하여 퍼블리싱을 할 때 절차 및 방법을 기술합니다. 아울러 개발 시 유의할 사항과 가이드라인 등을 다룹니다. 본 프로젝트의 개발 절차 및 방법을 가이드 함으로써, 개발의 생산성과 효율성을 향상하고, 개발된 결과의 통일성을 유지하여 향후 유지보수의 편의성, 높은 가독성을 제공하기 위함입니다.

1.2. 문서의 범위

본 문서에서는 다음의 사항을 기술합니다.

- 퍼블리싱 방식
- 컴포넌트 스타일링
- 프로그레스 스타일링
- 퍼블리싱 컨벤션 가이드
- 유의사항

1.3. 대상

본 문서는 Alopex 프레임워크를 활용하여 퍼블리싱을 하는 퍼블리셔를 대상으로 합니다.

2. 퍼블리싱 방식 결정

Alopex 컴퍼넌트 클래스를 사용할지 또는 새로운 컴퍼넌트 클래스를 정의할지 결정합니다.

2.1. Alopex 컴퍼넌트 클래스 사용

alopex-ui.css 컴퍼넌트 클래스를 사용하는 경우 기본 컴퍼넌트 클래스 내부에서 스타일을 재정의 하거나 확장하면 됩니다.

- 스타일 재정의 예제 코드

기본 컴퍼넌트 클래스인 .Button 내부에서 스타일을 재정의 합니다.

```
alopex-ui.css

.Button{
    …
    //customizing your css style
}

<button class="Button">basic</button>
```

©Copyright 2016 SK 주식회사 C&C. All rights reserved.

- 스타일 확장 예제 코드

기본 컴퍼넌트 클래스는 그대로 두고 확장을 할 수도 있습니다.

```
.Button.red{
    background: #ff0000;
}

<button class="Button red">basic</button>
```

2.2. 새로운 컴퍼넌트 클래스 정의

alopex-ui.css 컴퍼넌트 클래스를 사용하지 않고 새로운 컴퍼넌트 클래스를 정의하는 방법입니다. 아래와 같이 새로운 버튼 컴포넌트의 클래스를 새로 정의합니다.

```
.btn{
      //customizing your css style
}
<button class="btn">basic</button>
```

이 경우 $a.setup 함수를 사용하여 해당 클래스가 Alopex UI 컴퍼넌트로 변환되도록 설정이 필요합니다.

```
$a.setup({
    defaultComponentClass: {
        Button: 'btn',
            …
    }
});
```

3. 컴포넌트 스타일링

3.1. 테마 스타일링

Alopex 에서 제공하는 디폴트 스타일 외에 새로운 테마 작성 시 기존 class 를 재정의 하여 테마별로 다른 css 파일을 생성합니다.

```
.Button {//customizing your theme css style}
.Button:active {//customizing your theme css style }
```

3.2. 이벤트 스타일링

커서가 컴퍼넌트 위에 위치할 경우(hover) 또는 커서가 눌린 경우(pressed)의 스타일 정의는 :hover 또는 :active 같은 Pseudo Css 를 사용하여 정의합니다.

©Copyright 2016 SK 주식회사 C&C. All rights reserved.

```
//button active
.Button:active {
      …
}

//button hover
.Button:hover {
      …
}
```

3.3. 상태 스타일링

Alopex UI 컴퍼넌트 별로, 상태 값을 나타내는 클래스명이 동적으로 추가됩니다.

```
.Button.Checked {
…
}

.Button.Disabled {
      …
}
```

아래는 상태 종류별 설명입니다.

- .Checked
  - ✓ 컴퍼넌트의 check/uncheck 상태를 나타내는 클래스
  - ✓ 적용 컴퍼넌트: checkbox, radio
- .Disabled
  - ✓ 모든 입력 컴퍼넌트에 적용 가능하며, 컴퍼넌트가 비활성화 된 경우 추가됩니다.
  - ✓ 적용 컴퍼넌트: datepicker, button, dateinput, divselect, dropdown, radio, select, spinner, checkbox, textinput, textarea, tabs, accordion
- .Selected
  - ✓ 적용 컴퍼넌트: 페이징, 탭, 트리
- .Expandable
  - ✓ 로 구성된 마크업에서 하위 리스트가 존재하고, 하위 리스트에 아이템이 있을 경우 추가됩니다.
  - ✓ 적용 컴퍼넌트: dropdown, navmenu, tree, accordion
  - .Expanded
    - ✓ Expandable 상태에서 하위 리스트가 보여질 때 추가됩니다.
    - ✓ 적용 컴퍼넌트: dropdown, navmenu, tree, accordion

©Copyright 2016 SK 주식회사 C&C. All rights reserved.

4. 컨벤션 가이드

퍼블리싱을 하면서 html, css 코딩을 할 때 준수해야 하는 가이드 입니다. 퍼블리싱 결과의 웹 접근성 준수와 코드의 통일성을 유지하여 향후 유지보수의 편의성, 높은 가독성을 제공하기 위한 가이드라인입니다. 출처(http://codeguide.co/)

4.1. HTML 작성 가이드

4.1.1. Syntax

- indent: 탭(tab)을 사용합니다. 2 spaces 가 에디터별로 똑같은 가독성이 보장되긴 하지만, javascript 개발 시 탭(tab)을 사용하기 때문에 통일성 측면에서 탭을 사용합니다.
- nested 된 element 는 반드시 indent 를 줍니다.
- attribute 에는 double quotation(")를 사용합니다.
- Less Than(<), Greater Than(>), Ampersand(&) 기호에 HTML 엔터티를 사용합니다. (예>< > &)

```
<body>
    <img src="images/company-logo.png" alt="Company">
    <h1 class="hello-world">Hello, world!</h1>
</body>
```

4.1.2. doctype

doctype 은 html5 로 지정합니다.

```
<!DOCTYPE html>
```

4.1.3. language 속성

웹 접근성을 위하여 랭귀지를 명시합니다. 스크린리더가 읽을 때 필요합니다.

```
<html lang="ko">
```

4.1.4. IE 호환성

IE 최신버전 모드에서 작동하도록 아래 메타 태그를 작성합니다.

```
<meta http-equiv="X-UA-Compatible" content="IE=Edge">
```

©Copyright 2016 SK 주식회사 C&C. All rights reserved.

4.1.5. 인코딩

올바른 인코딩을 위해 반드시 명시합니다.

```
<meta charset="UTF-8">
```

4.1.6. CSS 와 인클루드

불필요한 type javascript 이나 type text/css 속성은 생략합니다. Type 속성을 사용해도 문법적으로 유효하나 생략하는 경우 기본값이 text/css 또는 text/javascript 로 처리가 되기 때문에 생략합니다. 단 부득이하게 doctype 이 HTML5 가 아닌 경우라면 type 속성은 생략할 수 없습니다.

```
<!-- External CSS -->
<link rel="stylesheet" href="code-guide.css">

<!-- In-document CSS -->
<style>
  /* ... */
</style>

<!-- JavaScript-->
<script src="code-guide.js"></script>
```

4.1.7. attr 순서

attribute 작성은 아래 순서로 합시다. id 보다 class 를 많이 쓰는 것이 가장 적절한 형태이기 때문입니다.

1. class
2. id, name
3. data-*
4. src, for, type, href, value
5. title, alt
6. aria-*, role

4.1.8. Boolean 속성

Boolean 속성에 optional 한 value 인 값(checked="checked")은 넣지 않습니다.. checked 로 명시합니다.

```
<input type="text" disabled>
<input type="checkbox" value="1" checked>
<select>
  <option value="1" selected>1</option>
</select>
```

©Copyright 2016 SK 주식회사 C&C. All rights reserved.

4.1.9. 마크업 줄이기

굳이 parent 가 필요 없는 경우라면 등을 과도하게 만들지 않습니다.

```
<!-- Not so great -->
<span class="avatar">
  <img src="...">
</span>

<!-- Better -->
<img class="avatar" src="...">
```

4.1.10. 동적 마크업

필요한 경우가 아니라면 javascript 로 동적으로 markup 을 만들지 않습니다.

4.2. CSS 작성 가이드

4.2.1. Syntax

- 탭은 탭(tab) 사용. (html syntax 와 동일 이유)
- 여러 셀렉터를 하나에 쓸 경우, 한 줄에 하나의 셀렉터 사용
- { 앞에 한칸 띄우기
- 닫는 } 는 새로운 줄에 기록
- : 다음에 한칸 띄우기
- 모든 선언은 각기 다른 줄에 기록
- 세미콜론(;)은 반드시 적기. (마지막 줄이라도 명시)
- 콤마(,) 다음에 한칸 띄우기
- rgb(), rgba(), hsl(), hsla(), rect() 안의 콤마(,)는 띄어쓰기 하지 않음
- 소수점 0 으로 시작하는 value 는 0 을 생략. 0.5 대신 .5 사용
- 속성 값이 '0'이면 단위를 생략.
- 색상값은 단축 표기하지 않음. (IE 의 Gradation 을 적용할 경우 색상값이 적용되지 않는 경우가 있음)

```
/* Bad CSS */
.selector, .selector-secondary, .selector[type=text]

/* Good CSS */
.selector,
.selector-secondary,
.selector[type="text"] {
```

©Copyright 2016 SK 주식회사 C&C. All rights reserved.

4.2.2. 선언 순서

선언의 순서는 아래와 같은 순서로 그룹핑 합니다.

1. 위치 관련
2. 박스모델 관련
3. 타이포그래피
4. 시각효과
5. 기타

```
.declaration-order {
  /* Positioning */
  position: absolute;
  top: 0;
  right: 0;
  bottom: 0;
  left: 0;
  z-index: 100;

  /* Box-model */
  display: block;
  float: right;
  width: 100px;
  height: 100px;

  /* Typography */
  font: normal 13px "Helvetica Neue", sans-serif;
  line-height: 1.5;

  color: #333;
  text-align: center;

  /* Visual */
  background-color: #f5f5f5;
  border: 1px solid #e5e5e5;
  border-radius: 3px;

  /* Misc */
  opacity: 1;
}
```

4.2.3. @import 사용 금지

CSS 내에 @import 는 성능을 저하시킵니다. 여러 개의 CSS 를 link 합니다.

```
<!-- Use link elements -->
<link rel="stylesheet" href="core.css">

<!-- Avoid @imports -->
```

©Copyright 2016 SK 주식회사 C&C. All rights reserved.

```
<style>
  @import url("more.css");
</style>
```

4.2.4. prefixed 속성

브라우저를 위한 -webkit-같은 prefix 를 사용할 경우 indent 를 속성에 정확히 맞춥니다.

```
.selector {
  -webkit-box-shadow: 0 1px 2px rgba(0,0,0,.15);
         box-shadow: 0 1px 2px rgba(0,0,0,.15);
}
```

4.2.5. Single declarations

CSS 의 선언은 하나의 선언 당 하나의 라인으로 구성합니다. 여러 선언을 하나의 라인으로 구성하면 디버깅하기 어렵습니다.

```
/* Single declarations on one line */
.span1 { width: 60px; }
.span2 { width: 140px; }
.span3 { width: 220px; }

/* Multiple declarations, one per line */
.sprite {
  display: inline-block;
  width: 16px;
  height: 15px;
  background-image: url(../img/sprite.png);
}
.icon          { background-position: 0 0; }
.icon-home      { background-position: 0 -20px; }
.icon-account   { background-position: 0 -40px; }
```

4.2.6. Shorthand notation

CSS 속성 중 일부(top, left 등)만 override 하기 위해서는 padding, background, border 와 같은 축약 표현(Shorthand notation)을 사용하지 않고 자세한 표현(Longhand notation)을 사용합니다.

```
/* Bad example */
.element {
  margin: 0 0 10px;
  background: red;
  background: url("image.jpg");
}

/* Good example */
.element {
  margin-bottom: 10px;
  background-color: red;
  background-image: url("image.jpg");}
```

©Copyright 2016 SK 주식회사 C&C. All rights reserved.

4.2.7. 주석

클래스 명을 또 다시 주석으로 만들 필요는 없습니다. 주석을 달 것이면 자세한 내용을 문장 형태로 명시합니다.

```
/* Bad example */
/* Modal header */
.modal-header {
  ...
}

/* Good example */
/* Wrapping element for .modal-title and .modal-close */
.modal-header {
  ...
}
```

4.2.8. 클래스명

- 클래스명은 소문자와 대시(-)만 사용(단 Alopex 의 클래스만은 구분을 위해 대문자로 시작)
- 과도한 약어를 사용하지 않음(.btn 은 괜찮지만, .s 와 같은 것은 아무 의미가 없음)
- 클래스명은 짧고 간결하게 작성
- 의미 있는 이름을 작성
- 가까운 부모나 기본 클래스에 기반하여 prefix 를 추가(예. .news-header, .news-footer)
- 클래스명의 첫번째 문자로 언더스코어나 숫자를 사용하지 않습니다.

```
/* Bad example */
/* Modal header */
.modal-header {
  ...
}

/* Good example */
/* Wrapping element for .modal-title and .modal-close */
.modal-header {
  ...
}
```

©Copyright 2016 SK 주식회사 C&C. All rights reserved.

4.2.9. 셀렉터

- 랜더링 성능을 최적화하기 위해 클래스(class)를 사용함.
- Attribute Selector 는 되도록이면 피함(성능 상 문제)
- 셀렉터의 수는 3 개정도까지만으로 제한
- 필요할 경우에만 바로 위 부모를 사용하여 클래스를 제한(prefix 를 사용하지 않을 경우에만 제한. prefix 를 사용하면 그럴 필요가 없음.)

```
/* Bad example */
span { ... }
.page-container #stream .stream-item .tweet .tweet-header .username { ... }
.avatar { ... }

/* Good example */
.avatar { ... }
.tweet-header .username { ... }
.tweet .avatar { ... }
```

4.3. 구조

4.3.1. 에디터 속성

- 탭은 2 space 대신에 Tab 을 사용
- 문장 맨 뒤의 공백 문자는 제거
- 파일은 UTF-8 로 인코딩
- 파일 마지막에 빈 줄 한 줄을 추가

©Copyright 2016 SK 주식회사 C&C. All rights reserved.