성능 가이드

개요

- eXBuilder6의 렌더링 엔진은 일반적인 렌더링 최적화 기법을 포함하고 있습니다.
- 다음의 상황들을 가급적 피하여 주십시오:
  - 그룹의 깊이를 너무 깊게 만들지 마십시오.
  - CSS에서 속성 연산 셀렉터를 가급적 사용하지 마십시오.
  - 특정 요소가 그려질 때 그 값이 계산의 결과 값인 경우를 가급적 피하십시오.

- `load` 이벤트에서 `redraw`가 필요한 초기화 작업은 `init`이벤트에서 처리하여 `redraw`회수를 줄일 수 있습니다.
- `DeferrredUpdateManager.INSTANCE.update()`는 즉시 그리기로 코드 흐름에서 UI가 갱신된 상태에서 값을 가져와서 작업하고 싶은 경우 사용합니다. 특별한 경우가 아니라면 `UIControl.redraw()`(데이터맵, 데이터셋에 바인딩된 경우 데이터셋의 값을 직접 변경하면 컨트롤을 redraw해서 UI를 갱신해야 함. 그리드만 자동으로 그리기 지원.) 사용을 권장합니다.

컨트롤 파퓰레이션

eXBulder6는 VirtualDOM 기술을 이용하여 그리기 작업을 자동으로 최적화 합니다.

```
for (var a=0; a

- 그룹에 컨트롤을 추가하는 경우 다음의 규칙을 준수하여 주십시오.
    - 여러 컨트롤을 추가하는 경우, 자식 컨트롤을 연속되는 위치에 추가하여 주십시오.
        - O - `grp1`의 2번째 3번째 4번째 자식으로 아웃풋을 추가.
        - X - `grp1`의 2번째 4번째 자식으로 아웃풋을 추가 - 불연속적인 위치에 대한 자식 추가는 성능 저하 요인이 됩니다.
    - 한 그룹으로부터 자식 컨트롤을 제거하는 코드와 새로운 자식 컨트롤을 추가하는 코드는 한 이벤트 틱 처리에서 실행되지 않도록 처리하여 주십시오.
    -


#### 강제 화면 업데이트

`DeferrredUpdateManager.INSTANCE.update()`는 현재 예약된 그리기 작업을 호출된 시점에 강제로 모두 실행하게 됩니다.
일반적으로 이러한 그리기 작업은 사용자에게 다음 프레임에 표시하기 직전까지 미리 실행될 필요가 없기 때문에, 자동적으로 지연(Deferred)됩니다.
```

직접적으로 이를 호출하는 것은 불필요한 성능 저하를 일으킬 수 있으므로, 지양하여 주십시오.
다음 그릴 DOM 상태를 현재 코드 내에서 예측해야 할 필요가 있고 다른 대안이 없는 경우에만 사용하여 주십시오.

화면이 갱신된 이후의 코드를 실행 하려면, 대신 `DeferrredUpdateManager.INSTANCE.asyncExec()`를 사용하여 작업을 예약 하십시오.