Jindo Reference 작성해 보기

2012.2.02 김진형

**$Jindo 클래스**

* 버전에대한 정보가 담겨 있는 클래스

**Properties**

**1.** object cl

cl 자신의 객체를 저장하는 프로퍼티

**$ 클래스**

* $를 이용하여 객체를 생성하고 조작할 수 있는 객체 형체로 리턴하는 역할을 한다.

**Properties**

1. object sID
   1. sID 는 돔에 생성할 요소들.
2. Array array
   1. 우의 돔 요소들을 저장할 배열

**$Class 클래스**

* 자바스크립트를 객체지향 언어로 구현할 수 있도록 지원하는 클래스

**Properties**

1. Object oDef
   1. 클래스의 생성자, 속서, 메서드등을 넣을 수 있다. 그 클래스를 정의할 수 있다.

**Method**

1. **$Class extend(superclass)**
   1. **부모 클래스를 상속 받는다**

**$A 클래스**

- 배열을 Array 로 리턴한다.

**Properties**

1. object cl
   1. cl 자신의 객체를 저장하는 프로퍼티

**Methods**

* 1. 생성자(Array array)
     1. 배열을 생성하는 배열 array 인자 입니다.
  2. String toString()
     1. Sting 형태로 변환해서 반환하는 메서드
  3. Boolean has()
     1. array 형태를 인자로 받을 수 있다.
  4. Number length(Number nLen, oValue)
     1. 배열의 길이를 지정, 반환
        1. Number nLen
           1. 지정한 nLen 만큼의 배열크기를 지정.
        2. Object oValue
           1. 배열의 크기가 늘어날 경우 oValue의 값을 늘어난 배열공간에 넣어준다.
  5. Number indexOf(Number|String|Object|Array oValue)
     1. 사용자가 원하는 특정값의 위치를 탐색하고 index를 반환한다. 없을 경우 -1 반환
        1. Object oValue
           1. 탐색할 원소를 저장하는 객체 입니다. 문자열, 숫자, 객체가 들어올 수 있다.
  6. Array $value
     1. 해당하는 배열을 반환한다.
  7. Number push(Number|String|Object|Array oValue, …….)
     1. 값을 배열에 넣어준다. 그 후 배열 length 값을 반환 (만약 arguments 와 같이 배열이 아닐 경우 배열형태로 바꾸어서 넣어준다)
  8. Variant pop()
     1. 해당 배열의 마지막 원소를 삭제 후 그 값을 반환
  9. Variant shift()
     1. 해당 배열의 맨 앞 원소는 삭제되고 한 칸씩 앞으로 당긴다.
  10. Number unshift(oValue1, …..)
      1. 해당 배열의 맨 앞쪽에 해당 원소(들) 을 추가한다. 여러 개를 한번에 입력할 경우 그 순서 그대로 들어간다.
  11. {$A} forEach(Function fCallback, Object oThis)
      1. 배열 내부의 원소를 돌면서 콜백함수를 실행한다.
         1. Function fCallback
            1. 콜백함수를 저장한다.
         2. Object oThis
            1. 콜백 함수가 객체의 메서드 일 떄 콜백 함수 내부에서 this 키워드의 실행문맥으로 사용할 객체
  12. {$A} slice(Number nStart, Number nEnd
      1. nStart 에서 nEnd 까지 의 부분을 빼온 후 [ ]로 래핑한 새로운 ${A}객체 형태로 반환한다.
         1. Number nStart
            1. 시작 위치
         2. Number nEnd
            1. 끝위치
  13. ${A} splice(Number nIndex, ,Number nHowMany, variant oValues, …..)
      1. 배열의 nIndex 위치부터 nHowMany 반큼 삭제한후 oValues들을 추가할 수 있고 ${A} 형태로 리턴한다.
         1. Number nIndex,
            1. 시작 인덱스 숫자
         2. Number nHowMany
            1. 시작 인덱스부터 nHowMany 까지 삭제
         3. Variant oValues
            1. 추가할 값들 지정한다.
  14. ${A} shuffle()
      1. 배열 원소들의 순서를 Random 하게 섞는다.
  15. ${A} reverse()
      1. 해당 배열 원소들의 순서를 뒤집는다
  16. {$A} empty()
      1. 해당 배열을 비운다. (길이를 0 으로 만든다.)
  17. $A.Break() 🡺 forEach(), filter(), map() 메서드 루프 중단
      1. 강제적으로 에러를 발생 시켜서 중단시키는 메서드
  18. $A.Continue()
      1. 내부적으로 강제로 에럴 발생시켜 다음 루프로 건너 뛰게 함
  19. ${A} map(fCallback, oThis)
      1. 콜백 함수를 실행하고 그것의 결과 값을 배열에 저장한다.
         1. Function fCallback
            1. Value,index,array를 갖는 함수
         2. Object oThis
            1. 콜백 함수가 객체의 메서드일 때 콜백 함수 내부에서 this 키워드의 실행문맥 사용할 객체이다.
  20. ${A} filter(fCallback, oThis)
      1. 해당 함수를 실행 결과 얻은 결과 값을 모아서 ${A} 객체를 리턴 한다.
         1. Function fCallback
            1. Value,index,array를 갖는 함수
         2. Object oThis
            1. 콜백 함수가 객체의 메서드일 때 콜백 함수 내부에서 this 키워드의 실행문맥 사용할 객체이다.
  21. Boolean every(fCallback, oThis)
      1. 배열의 모든 원소를 forEach로 순회 하면서 fCallback 함수의 조건과 맞는지 보고 맞으면 true를 반환하고, 하나라도 틀리면 순회를 중지하고 false를 즉시 반환한다.
         1. 변수는 위와 동일
  22. Boolean some(fCallback, oThis)
      1. 배열의 모든 원소를 순회하면서 fCallback 함수에서 반환된 조건과 같은지 확인한 후 하나라도 같은게 있으면 true 를 반환하고, 없으면 false를 반환한다.
  23. ${A} refuse(Object oValues)
      1. 해당 oValues인자들의 값을 제외하고 ${A} 객체를 만들어 반환한다.
  24. ${A} unique
      1. 중복되는 원소를 제거하여 ${A} 객체를 반환한다.

**$Element 클래스**

- $Element 객체는 해당하는 html 을 래핑 하여 좀더 쓰기 편하게 해준다

**Properties**

1. **Jindo.$Element cl**
   1. **본인의 객체를 저장하는 프로퍼티**

**Methods**

1. 생성자($Element(Element))
   1. $Element 객체를 생성한다.
2. $Element $value()
   1. 원래의 Element 가 반환된다.
3. {$Element} visible(bVisible)
   1. 해당 엘리먼트의 Css display 속성을 none 혹은 show로 지정해 준다. {$Element} 의 visible 속성을 변경한 객체를 반환한다.
      1. Boolean bVisible
         1. True 인경우 display를 속성을 보이게 설정 false인 경우 none이라고 설정
   2. ${Element} show()
      1. 해당 엘리먼트의 속성을 보이게 display 설정
         1. 태그별로 block 값을 지정해 놓았다. 지정된 값이 없으면 inline 으로 설정해서 객체를 반환한다
            1. String s

해당 ${Element}의 style

* + - * 1. Array C

태그별로 block 설정할 명령이 입력해 놓음

* 1. {$Element} hid()
     1. 해당 엘레먼트의 display 속성을 none 으로 바꾼다
  2. {$Element} toggle()
     1. 해당 엘레먼트의 display 속성을 hide 와 show를 번갈아 가면서 바꿔준다.
  3. Number| opacity(Number value)
     1. 해당 엘리먼트의 opacity(투명도) 값을 사용자가 지정한 value값으로 바꾼다. 값이 설정이 되어 있지 않을 때는 해당하는 값을 반환해 온다
        1. Number value
           1. 값을 지정
  4. String|${Element} Css(String sName,Number sValue)
     1. 해당 엘리먼트의 css 속성값을 가져오거나 설정한다.(카멜 표기법 사용)
        1. String sName
           1. Css 속성명
        2. Number sValue
           1. Css 속성값
  5. ${Element}| \_getCss, ${Element} \_setCss
     1. 내부적으로 CSS값을 설정하고 얻어 올떄 사용한다,
  6. String|${Element} attr(sName, sValue)
     1. 해당 엘리먼트의 속성을 가져오거나 설정한다. sName만 사용하면 속성값을 반환한다. 값까지 설정하면 해당 속성값을 설정한다.
        1. Object|String sName
           1. 속성이름을 지정
        2. Object|Numbe|String sValue
           1. 속성 값을 지정한다.
  7. Number|${Element} width(Number width)
     1. Width 값을 설정하면 그 값으로 해당 엘레먼트의 width 값 설정하고 해당 엘레먼트 객체를 받아온다. 없으면 offsetWidth 값을 받아온다.
        1. Number width
           1. 너비 값을 입력 받는다.
  8. Number|${Element} height(Number width)
     1. height 값을 설정하면 그 값으로 해당 엘레먼트의 width 값 설정하고 해당 엘레먼트 객체를 받아온다. 없으면 offsetHeight 값을 받아온다.
        1. Number height
           1. 높이 값을 입력받는다.
  9. String|${Element} className(Sting sClass)
     1. 해당 엘레먼트의 클래스 네임을 설정하거나 반환한다
        1. String sClass
           1. Optional 이고 클래스 네임을 지정받는다.
  10. Boolean hasClass(sClass)
      1. 해당 엘리먼트에서 sClass 가 사용되고 있는지 알아보고 사용된다면 true, 안된다면 false를 반환한다
         1. String sClass
            1. 찾아볼 클래스명을 입력한다.
  11. ${Element} addClass(sClass)
      1. 해당 엘리먼트에서 sClass가 있는지 찾아보고 있으면 엘리먼트를 리턴시키고 없다면 해당 클래스을 추가한다.
         1. String sClass
            1. 찾아볼 지정할 클래스명 입력한다.
  12. ${Element} removeClass(sClass)
      1. 해당 엘리먼트에 sClass 라는 클래스를 제거한다.
         1. String sClass
            1. 제거할 클래스명 입력한다
  13. ${Element} toggleClass(sClass, sClass2)
      1. sClass 와 sClass2 를 번갈아 가면서 스위치처럼 토글 시킨다. 즉 해당 클래스명의 엘리먼트를 반환시킨다.
         + 1. String sClass,sClass2

클래스명을 입력한다.

* 1. ${Element}|String text(String sText)
     1. 해당 엘리먼트의 텍스트 노드 값을 설정하거나 가져온다.
        1. String sText
           1. 입력할 텍스트 값
  2. ${Element}|String html(sHTML)
     1. 해당 엘리먼트 내부의 내용을 모두 가져오거나, 값을 설정하면 그값으로 변경한다.
        1. String SHTML
           1. 변경할 text
  3. String outerHTML()
     1. 해당 엘리먼트의 본인코드 부분과 내부 innerHTML 을 함께 String 형태로 반환한다.
  4. String toString()
     1. OuterHTML() 과 같다
  5. {$Element} apper(duration, callback)
     1. 해당 엘리먼트의 요소들을 duration 시간 만큼 설정해서 서서히 보이게 해준다.
        1. Number duration
           1. 완전히 나타낼 때 까지 시간
        2. Funtion callback
           1. 해당 엘리먼트가 완전히 나타난후 실행할 콜백함수
  6. {$Element} disapper(Number duration, Function callback)
     1. 해당 엘리먼트의 요소들을 duration 시간 만큼 설정해서 서서히 보이게 해준다.
        1. Number duration
           1. 완전히 없어질 때 까지 시간
        2. Funtion callback
           1. 해당 엘리먼트가 완전히 사라진후 실행할 콜백함수
  7. ${Element} EvalScripts(String sHTML)
     1. Shtml에 지정한 문자열을 자바스크립트 코드로 실행 할 수 있게 해준다.
        1. String Shtml
           1. 스트링형태의 자바스크립트 구문을 입력한다.
  8. ${Element} append(${Element|String oElement)
     1. 해당 엘리먼트에 입력받은 엘리먼트를 그 자신의 마지막 자식 노드에append시킨다.
        1. oElement
           1. 해당 요소를 자식으로 append 시킨다.
  9. ${Element} prepend(${Element|String oElement)
     1. 해당 엘리먼트에 입력받은 엘리먼트를 그 자신의 마지막 자식 첫번째노드에prepend시킨다.
        1. oElement
           1. 해당 oElement 요소를 첫번째 자식으로 append 시킨다.
  10. ${Element} appendTo(${Element|String oElement)
      1. 해당 객체를 입력받은 oElement 객체의 마지막 자식요소로 통째로 집어 넣는다.
         1. oElement
            1. 해당 oElement 요소를 마지막 자식으로 append 시킨다.
  11. ${Element} prependTo(${Element|String oElement)
      1. 해당 객체를 입력받은 oElement 객체의 첫번째 자식요소로 통째로 집어 넣는다.
         1. oElement
            1. 해당 oElement 요소를 첫번째 자식으로 append 시킨다.
  12. ${Element} replace(${Element|String oElement) 🡸v
      1. 해당 엘리먼트에 지정한 oElement로 치환시킨다.
         1. oElement
            1. 해당 oElement 값으로 치환시킨다.
  13. ${Element} before(${Element|String oElement)
      1. oElement 객체를 해당 엘리먼트의 바로 이전 형제노드로이동
         1. ${Element|String oElement
            1. 이동시킬 oElement 객체
  14. ${Element} after (${Element|String oElement)
      1. oElement 객체를 해당 엘리먼트의 바로 다음형제노드로이동
         1. ${Element|String oElement
            1. 이동시킬 oElement 객체
  15. ${Element}|Array parent(Functino pFunc,Number limit)
      1. 해당 객체의 바로위 부모 요소들을 찾아준다
         1. Function pFunc
            1. 해당 콜백 함수의 결과가 true를 반환하는 경우 에만 배열에 저장 되고 그 콜백에 들어가는 내용은 parentNode의 엘리먼트 객체가 각각 입력되게된다.
         2. Number limit
            1. 탐색할 깊이 레벨 값
  16. ${Element}|Array child(Functino pFunc,Number limit)
      1. 해당 객체의 바로 밑에 자식 요소들을 찾아준다
         1. Function pFunc
            1. 해당 콜백 함수의 결과가 true를 반환하는 경우 에만 배열에 저장 되고 그 콜백에 들어가는 내용은 parentNode의 엘리먼트 객체가 각각 입력되게된다.
         2. Number limit
            1. 탐색할 깊이 레벨 값
  17. ${Element}|Array prev(Function pFunc)
      1. 해당 엘리먼트의 이전 형제 노드에 해서 pFunc값을 줘서 해당 조건에 만족하는 객체 엘리먼트 리턴, 혹은 콜백 함수가 없을 경우 그냥 검색
         1. Function pFunc
            1. 찾을 엘리먼트에 적용할 콜백 함수.
  18. ${Element}|Array next(Function pFunc)
      1. 해당 엘리먼트의 다음 형제 노드에 해서 pFunc값을 줘서 해당 조건에 만족하는 객체 엘리먼트 리턴, 혹은 콜백 함수가 없을 경우 그냥 검색
         1. Function pFunc
            1. 찾을 엘리먼트에 적용할 콜백 함수.
  19. ${Element} first()
      1. 해당 엘리먼트의 자식 노드에서 첫번째에 해당하는 엘리먼트만 반환시킨다
  20. ${Element} last()
      1. 해당 엘리먼트의 자식 노드에서 마지막에 해당하는 엘리먼트만 반환시킨다
  21. Boolean isChildOf(${Element}|String element)
      1. 해당 엘리먼트가 element 인자로 들어온 엘리먼트의 자식인지 확인하는 메서드 맞으면 true 없으면 false 값을 반환한다.
         1. ${Element}|String element
            1. 엘리먼트나 id값을 입력 받는다.
  22. Boolean isParentOf(${Element}|String element)
      1. 해당 엘리먼트가 element 인자로 들어온 엘리먼트의 부모인지 확인하는 메서드 맞으면 true 없으면 false 값을 반환한다.
         1. ${Element}|String element
            1. 엘리먼트나 id값을 입력 받는다.
  23. Boolean isEqual((${Element}|String element)
      1. 해당 엘리먼트가 인자로 들어도 element 값과 같은지 확인 한후 같으면 true를 리턴한다.
         1. ${Element}|String element
            1. 엘리먼트나 id값을 입력 받는다.
  24. ${Element} fireEvent(String sEvent,object oProps)
      1. 해당 엘리먼트에 sEvent가 발생하면 oProps의 속성을 이용할 수 있다.
         1. String sEvent
            1. 이벤트 이름을 입력
         2. object oProps
            1. 이벤트 발생시킬 때 속성 설정
  25. ${Element} empty()
      1. 해당 엘리먼트와 그 내부에 있는 html 을 제거 한다. 그리고 엘리먼트 객체를 반환시킨다.
  26. ${Element} remove(${Element}|String oChild)
      1. 해당 엘리먼트의 자식노드중에 oChild와 같은 것을 제거한다.
         1. (${Element}|String oChild
            1. 삭제할 자식 노드의 파라미터
  27. ${Element} leave()
      1. 해당 엘리먼트의 바로 부모노드에서 제거한 엘리먼트 객체만 반환
  28. ${Element} wrap(wrapper)
      1. 해당 엘리먼트를 wrapper로 감싸주고 그 전체를 리턴 시킨다.
  29. Number indexOf(element)
      1. Element 인자값이 해당 엘리먼트에서 몇번째 자신인지? 반환값은 없으면 -1 있으면 Number 값이 리턴된다.

**$Fn 클래스**

- $Fn 객체는 Function을 래핑하여 쉽게 함수를 사용 할 수 있도록 도와준다.

**Properties**

1. **Jindo.$Fn cl**
   1. **본인의 객체를 저장하는 프로퍼티**

**Methods**

1. 생성자($Fn(Element))
2. ${Fn} $value()
   1. 원래 ${Fn} 객체를 반환 시켜준다
3. Function bind()
   1. 생성자가 지정한 객체의 메서드로 동작할 수 있도록 묶은 Function객체를 반환한다. apply함수를 이용해서 내부적으로 binding 시켜 준다.
      1. Function arguments들
         1. Function을 등록한다.
4. Function bindForEvent()
   1. 이벤트를 위해서 함수를 바인딩 한다. apply함수를 이용하여 범위를 지정해 준다.
5. ${Fn} attach(oElement, String sEvent)
   1. 함수를 이벤트 핸들러로 등록하는 메서드 이다.
      1. ${Element} oElement
         1. 이벤트 핸들러를 등록할 엘리먼트들
      2. String sEvent
         1. 등록할 이벤트
6. ${Fn} detach(oElement, String sEvent)
   1. 해당 엘리먼트에 등록된 이벤트 핸들러를 등록 해제 하는 메서드.
      1. ${Element} oElement
         1. 이벤트 핸들러를 등록할 엘리먼트들
      2. String sEvent
         1. 등록할 이벤트
7. ${Fn} delay(Number nSec,Array args)
   1. 해당하는 함수를 nSec 초 이후에 호출해서 쓸 수 있게 한다.
      1. Number nSec
         1. 딜레이 할 초
      2. Array args
         1. 파라미터 값들
8. ? setInterval(Number nSec, Array args)
   1. ${Fn} 객체를 일정 시간 마다 호출할 때 사용하는 메서드
      1. Number nSec
         1. 호출 간격 시간(초)
      2. Array args
         1. 파라미터 값들
9. Free(${Element} Element)
   1. 해당 엘리먼트의 이벤트를 제거 해준다
      1. Element
         1. 해당 엘리먼트 객체
10. \_domready(doc, Function func)
    1. 내부적으로 사용하는 함수이다. 돔이 완전이 불러질지면 그때 알려주는 역할을 한다. 돔이 준비가 되어야 엘리먼트를 찾고 더 나아가 이벤트를 할당 할 수 있기 때문이다.
       1. Function func
       2. Doc
11. Function \_fireWhenElementBoundary(String doc,Function func)
    1. 해당 엘리먼트의 바운더리에 갈 경우 function을 실행시킨다.
       1. String doc
          1. 더 찾아 볼 필요가 잇다.
       2. Function func
          1. 실행 할 함수
12. Void gc()
    1. ${Fn}을 통해서 등록된 모든 이벤트 핸들러를 해제한다,
13. void freeElement(Element oElement)
    1. oElement에 지정된 이벤트 핸들러를 모두 제거한다.
       1. Element oElement
          1. 삭제할 이벤트 핸들러가 등록되어 있는 엘리먼트