INSTITUT
TEKNOLOGI
HARAPAN
BANGSA
School of Telematics

Desain Pemrograman Web
IF-1P03

jQuery

Ken Ratri. MT

jQuery Effects - Animation

- The jQuery animate() method is used to create custom animations.
- The required params parameter defines the CSS properties to be animated.
- The optional speed parameter specifies the duration of the effect. It can take the following values: "slow", "fast", or milliseconds.
- The optional callback parameter is the name of a function to be executed after the animation completes.

Sintaks :

$(*selector*).animate({*params*},*speed*,*callback*);

```
$("button").click(function(){
  $("div").animate({left:'250px'});
});
```

jQuery animate() - Manipulate Multiple Properties

```
$("button").click(function(){
  $("div").animate({
    left:'250px',
    opacity:'0.5',
    height:'150px',
    width:'150px'
  });
});
```

jQuery animate() - Using Relative Values

it is also possible to define relative values (the value is then relative to the element's current value). This is done by putting += or -= in front of the value:

```
$("button").click(function(){
  $("div").animate({
    left:'250px',
    height:'+=150px',
    width:'+=150px'
  });
});
```

jQuery animate() - Pre-defined Values
s

You can even specify a property's animation value as "show", "hide", or "toggle":

```
$("button").click(function(){
  $("div").animate({
    height:'toggle'
  });
});
```

jQuery animate() - Uses Queue Functionality

By default, jQuery comes with queue functionality for animations.
This means that if you write multiple animate() calls after each other, jQuery creates an "internal" queue with these method calls. Then it runs the animate calls ONE by ONE.
So, if you want to perform different animations after each other, we take advantage of the queue functionality:

```
$("button").click(function(){
  var div=$("div");
  div.animate({height:'300px',opacity:'0.4'},"slow");
  div.animate({width:'300px',opacity:'0.8'},"slow");
  div.animate({height:'100px',opacity:'0.4'},"slow");
  div.animate({width:'100px',opacity:'0.8'},"slow");
});
```

jQuery Stop Animations

- The jQuery stop() method is used to stop animations or effects before it is finished.
- The stop() method works for all jQuery effect functions, including sliding, fading and custom animations.

```
$(selector).stop(stopAll,goToEnd);
```

```
$("#stop").click(function(){
  $("#panel").stop();
});
```

jQuery Callback Functions

- A callback function is executed after the current effect is 100% finished.
- syntax: **$(selector).hide(speed,callback)**
- **Examples**
- The example below has a callback parameter that is a function that will be executed after the hide effect is completed:

```
$("button").click(function(){
  $("p").hide("slow",function(){
    alert("The paragraph is now hidden");
  });
});
```

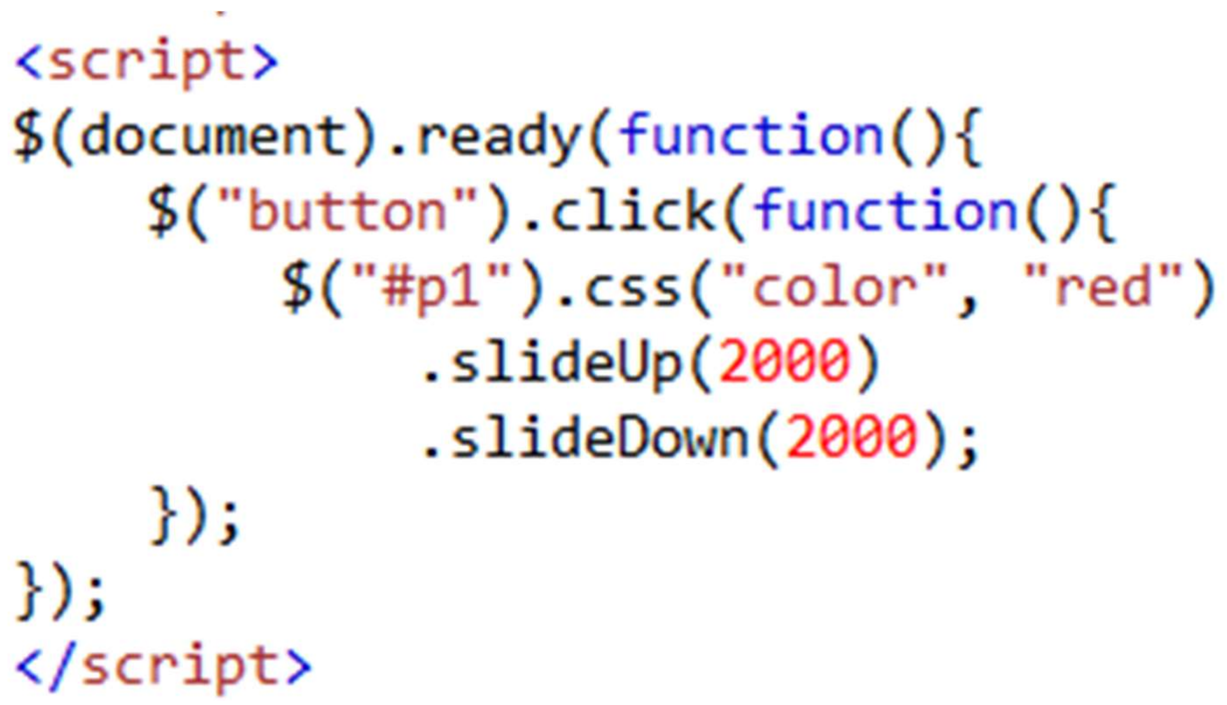
```
$("button").click(function(){
  $("p").hide(1000);
  alert("The paragraph is now hidden");
});
```

jQuery - Chaining

- With jQuery, you can chain together actions/methods.
- Chaining allows us to run multiple jQuery methods (on the same element) within a single statement.
- The following example chains together the css(), slideUp(), and slideDown() methods. The "p1" element first changes to red, then it slides up, and then it slides down:

```
$("#p1").css("color", "red").slideUp(2000).slideDown(2000);
```

jQuery - Chaining



```
<script>
$(document).ready(function(){
    $("button").click(function(){
        $("#p1").css("color", "red")
            .slideUp(2000)
            .slideDown(2000);
    });
});
</script>
```