WEEK 2

SUMMARY PRESENTATION
GENERIC!!
- Timelapse tool which search for the change in the world overtime.
- For example ice/land/forest/cities
- How to find a place? Name of the place/forest/city? Coordinates?

PROOF OF CONCEPT -> POSSIBLE QUESTIONS

- How does the tool exactly work?
- What does the interface look like?
- In which form will the result be shown?
- How will the outcomes be arranged?
- What does it look like if there is no outcome from your search?
- What does the homepage look like?
- Tips for searching like autocompletion?

Concept & tool questions

From our interest of Google maps, we where inspired to see how the earth changed over the last past decades, thanks to new technology we can now see what the impact is of human activity and the development on nature. We found it interesting to see, that the positive side and the negative side is viewable know days.

We love that the earth is suddenly an interactively explorable library. As designers we want to see what the boundaries are of this aspect. Some things are made clear, like climate changes, where some areas shrink or others expand. We as young designers see how important it is to be aware of change, we must be aware of what is changing and how we evolve with it, or what isn't.

For who is it designed?

Our scraping tool give the user the experience of time, it shows the user the change over a period of time. The scraping tool is not based on a specific users, or with a specific use in case, it is designed to use the most common denominator, so the use case that will benefit the most users.

How does the tool work?

The user types a coordinate into a search bar, that that will show you after you press enter a series of pictures that is shows you the change over time. After seeing the outcome, you are able to download the showed information.

What is the input?

The user has to put a in a specific coordinate that will provide the user to see a specific area. That also means that the user is free to put in whatever coordinate that the user wants to use and in that case the area is unknown and provides the user to see unknown or unexplored places.

What is the outcome?

The outcome of the tool will show you a specific area that shows you a change over a period of time. It will show you a specific outcome that is ordered in a certain chronological order.
In case that the user puts a wrong or non- excising coordinate into the search bar, it will tell you that this coordinate isn't availed on this earth in perhaps sometimes a not really expected way.

How does the interface look like?

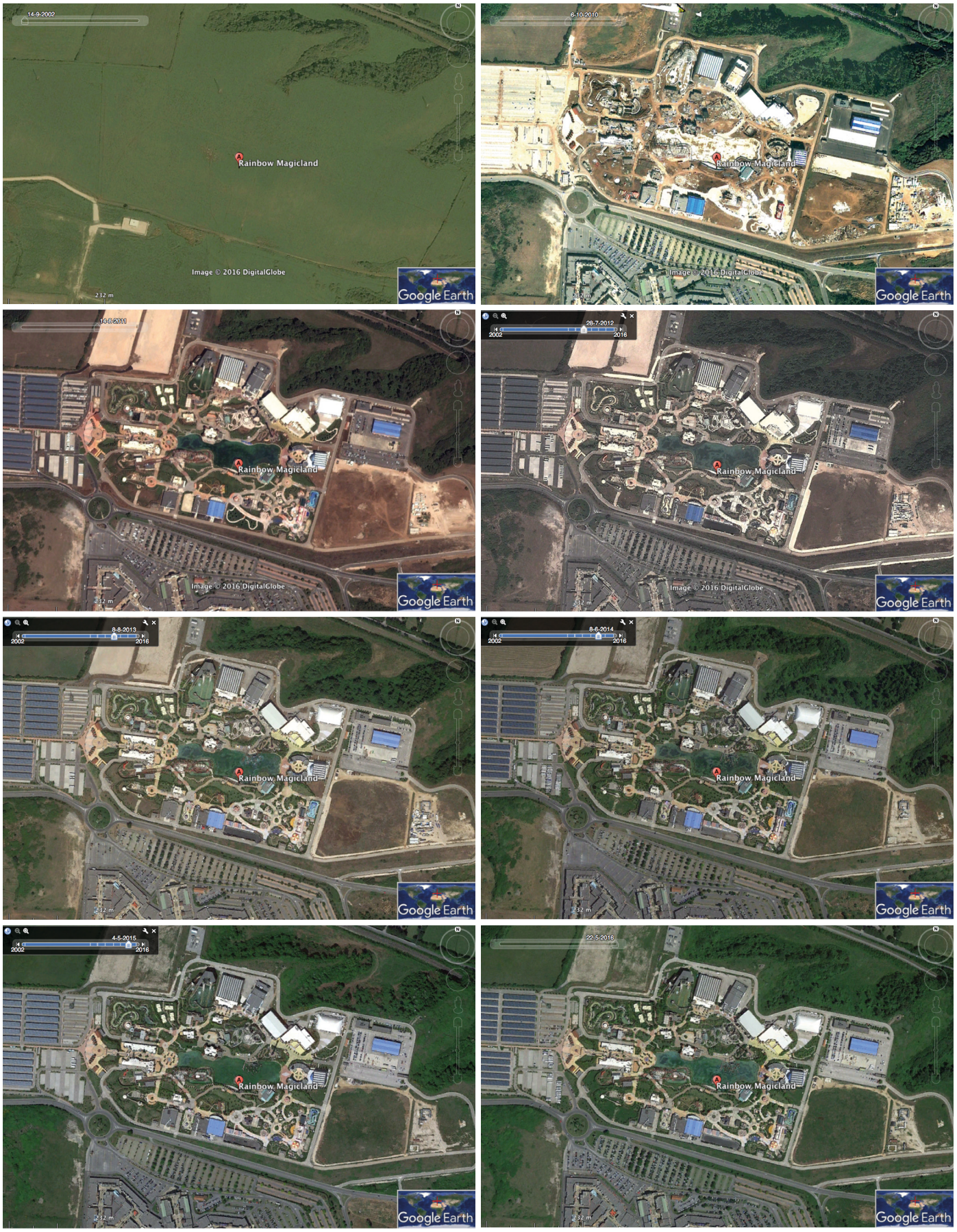
The interface is inspired on Google; simple to use.
There is a search bar where the user can put his coordinate in, and the interface is a friendly use system where the user quickly can see what he can do with the showed scraped information. It can be downloaded, or he can put a new coordinate into the tool, and start again with the process.

Who are involved?

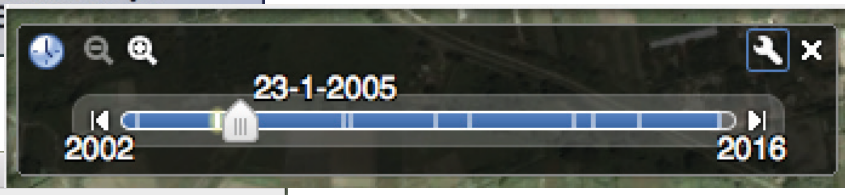
Carmen Steenbrink
Iris de Graaf
Marleen van Zalm
Google earth (engine)
Time laps

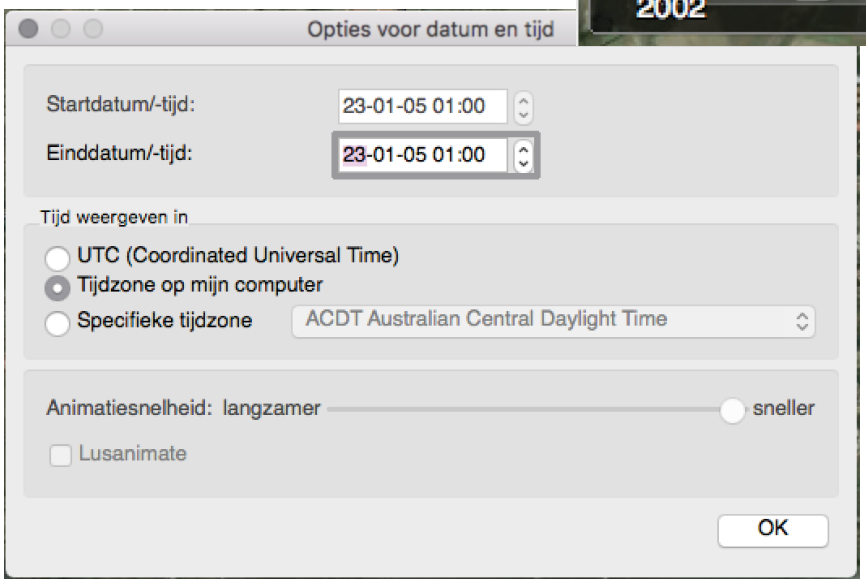
How does the tool exactly work?

Rainbow Magicland, Via della Pace, Valmontone, Rome, Italië



Options in google earth to manage time





What does the interface look like?

What does it look like if there is no outcome from your search?

Tips for searching like autocompletion?

Creating the Page

A good first step is to download a copy of the jQuery Autocomplete plugin. I am using the minified version within my header, along with a copy of the latest jQuery script. Since I am keeping all the suggestions inside a JavaScript array I have also moved the custom scripts into another separate file.

```
<!doctype html>
<html lang="en-US">
<head>
  <meta http-equiv="Content-Type" content="text/html;charset=utf-8">
  <title>Input Autocomplete Suggestions Demo</title>
  <link rel="shortcut icon" href="http://designshack.net/favicon.ico">
  <link rel="icon" href="http://designshack.net/favicon.ico">
  <link rel="stylesheet" type="text/css" media="all" href="style.css">
  <script type="text/javascript" src="js/jquery-1.9.1.min.js"></script>
  <script type="text/javascript" src="js/jquery.autocomplete.min.js"></script>
  <script type="text/javascript" src="js/currency-autocomplete.js"></script>
</head>
```

The resources are not too difficult to get setup. And similarly all of the HTML is very straightforward so you can implement any type of search design in your page and it should work fine. I am using an outer container with the ID **#searchfield** to center the whole form. There is no submit button for this demonstration, however the results will appear automatically in the dynamic context field below the input.

```
<div id="content">
  <h1>World Currencies Autocomplete Search</h1>
  <p>Just start typing and results will be supplied from the JavaScript. Check out <a href="

  <div id="searchfield">
    <form><input type="text" name="currency" class="biginput" id="autocomplete"></form>
  </div><!-- @end #searchfield -->

  <div id="outputbox">
    <p id="outputcontent">Choose a currency and the results will display here.</p>
  </div>
</div><!-- @end #content -->
```

We can target the field by using an ID of **#autocomplete**. There is no extra HTML because all of the suggestions will be presented using JS codes. One other interesting area is the paragraph using an ID of **#outputcontent**. This is where we can display the user's choice once they click on a suggestion. Normally on a live website you would instead just redirect the user to a search page, or onto the main page itself.

Designing with CSS

I do not want to delve too far into detail, but we should look at the CSS styles for each of the suggestion features, along with the main input area. The outer box border and content styles are heavily developed around the Design Shack search bar. Each of the suggestion styles is based off the default code, which I have copied over into my stylesheet.

```
.autocomplete-suggestion { padding: 10px 5px; font-size: 1.2em; white-space: nowrap; overflow
.autocomplete-selected { background: #f0f0f0; }
.autocomplete-suggestions strong { font-weight: normal; color: #3399ff; }
```

If you want to keep these in a separate stylesheet it would still load just fine. However I try to combine resources so that we may cut back on HTTP requests from the browser. The first two classes are targeting the entire suggestions popup, along with each internal suggestion row. You may style them in any way that fits your website. Additionally the CSS comes along with a selected feature class to highlight text which has already been typed by the user.

```
#searchfield { display: block; width: 100%; text-align: center; margin-bottom: 35px; }

#searchfield form {
  display: inline-block;
  background: #eeefed;
  padding: 0;
  margin: 0;
  padding: 5px;
  border-radius: 3px;
  margin: 5px 0 0 0;
}
#searchfield form .biginput {
  width: 600px;
  height: 40px;
  padding: 0 10px 0 10px;
  background-color: #fff;
  border: 1px solid #c8c8c8;
  border-radius: 3px;
  color: #aeaeae;
  font-weight:normal;
  font-size: 1.5em;
  -webkit-transition: all 0.2s linear;
  -moz-transition: all 0.2s linear;
  transition: all 0.2s linear;
}
#searchfield form .biginput:focus {
  color: #858585;
}
```

The search field itself should not be a problem regardless of your own design. jQuery Autocomplete is built to work around any width so that all suggestions can fit nicely into the same layout. Granted my example does take up a lot of space on the page, but the plugin handles this flawlessly without very many customizations.

Implement the Autocomplete

If you are using a backend Ajax call to fetch data results then you may have to perform your own search feature. jQuery Autocomplete expects some type of response data like XML/JSON and this should be returned from PHP, Rails, Python, ASP.NET, etc. So the most likely scenario is to search through your database table of entries and pull out all the related posts and their URLs, then return this list to your browser.

What I have done is created an array of entries right inside JavaScript for the major currencies found all around the world. As you start typing the country or currency name you will see a list of suggested items pulled directly from the array. It takes a long time for manual data entry, so I will only copy a small portion of the array to demonstrate how it is done.

```
$(function(){
  var currencies = [
    { value: 'Afghan afghani', data: 'AFN' },
    { value: 'Albanian lek', data: 'ALL' },
    { value: 'Algerian dinar', data: 'DZD' },
    { value: 'European euro', data: 'EUR' },
    { value: 'Angolan kwanza', data: 'AOA' },
    { value: 'East Caribbean dollar', data: 'XCD' },
...
    { value: 'Vietnamese dong', data: 'VND' },
    { value: 'Yemeni rial', data: 'YER' },
    { value: 'Zambian kwacha', data: 'ZMK' },
    { value: 'Zimbabwean dollar', data: 'ZWD' },
  ];
```

All of this content must be added before running your own function within jQuery. I have chosen to keep the array and custom function call stored within the same JS file. However you can choose to separate functions from data and move them into different files altogether. Now the autocomplete plugin only requires a small amount of default code. I will copy a sample template along with my live demo code so you can see what I've done using the same parameters.

```
  $('#autocomplete').autocomplete({
    lookup: currencies,
    onSelect: function (suggestion) {
    // some function here
    }
  });



  // setup autocomplete function pulling from currencies[] array
  $('#autocomplete').autocomplete({
    lookup: currencies,
    onSelect: function (suggestion) {
      var thehtml = '<strong>Currency Name:</strong> ' + suggestion.value + ' <br> <strong>S
      $('#outputcontent').html(thehtml);
    }
  });
```

The **lookup** parameter is only used for local data when we are targeting an array. **onSelect** will trigger a new function every time the user selects a suggestion. In my live demo we simply create a new string of HTML to output the selected name and value of that entry. But please notice that my demo is a very bare-bones example of what is possible.

Check out the jQuery Autocomplete documentation to get a better idea of some other parameters. I have not even begun to scratch the surface and JavaScript developers can likely put together examples using many other params in the function call. But I want to show how easy it is to get this plugin up and running on your website – even using static data where necessary.