Assignment 9 (HF, major subject)

Due: Wed 11.01.2017; 20:00h (3 Weeks)

Goals

After doing these tasks...

- you can apply watermarking techniques practically
- you can use steganography tools
- you are trained in advanced routing with Express

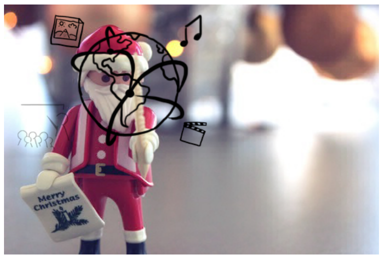
Task 1: Watermark Proxy

Difficulty: Medium

Update the watermarker module we provided on GitHub. The goal is to allow the user to visit a certain URL to create watermarked versions of images.

If the user visits the URL http://localhost:3000/watermark?url=https://images.pexels.com/photos/253366/pexels-photo-253366.jpeg?h=350 - a watermarked version of the image is shown.

➔



Here's what you have to do to solve this task:

a) Register a new route '/watermark' in routes/watermarker.js
b) Read the image URL from the query parameters by accessing req.query.url
c) Use Jimp to load the image, render the watermark and save the watermarked image.
d) Send the watermarked file with res.sendFile() or res.download()
e) If an error occurs during this process, send a nice placeholder image. You can use sorry.png from the skeleton files on GitHub for this assignment.

Submit the updated watermarker.js file.

Task 2: Steganography

Difficulty: Medium

In the watermarking example from the tutorial (and also Task 1), the watermarks are always visible. However, it is sometimes preferable to create hidden watermarks. If the watermark should be "secret", steganography can help. It is a technique to conceal information inside another medium.

Multiple steganography packages are available for NodeJS. We recommend stegosaurus.

Allow a user to interactively hide messages inside an arbitrary image. We provide a couple of files for this task to help you get started.

a) steganographer.js: this file is the main skeleton for this task. Extend / modify the code where it is marked by TODO items.

   Its main purpose is to register two routes:

   a. `/hide` takes one parameter "message" either via POST or GET. If POST is used, the message is in req.body.message, if GET is used the message is in req.query.message.

      Encode the message inside the image (variable `hostImage`) unsing the stegosaurus module's encodeString method.

      Generate a random image name for the output. We suggest using the uuid module, with a call to uuid.v1() or uuid.v4().

      Do not respond with the image directly. Instead respond with a JSON like this:

```
{
  id: generated_id,
  path: '/stegano/watermarked/' + generated_id + '.png',
  size: message.length
}
```

   b. `/seek` takes two parameters "id" and "size". It looks up the image with the corresponding id and decodes "size" number of bytes. Then it sends a JSON like this:

```
{ status: 'successfully decoded message',
  message: hiddenMessage }
```

b) index.html: Contains a fully working front-end to allow the user to enter secret messages, save the images with the hidden messages and decode their message. Feel free to modify this file to suit your needs.

c) merry-x-mas.png is a sample .png file that you can use to store the messages.

Add the steganographer module to the app (require it in app.js and register it with app.use).

You can add the functionality to the WatermarkApp from tutorial 09 (see GitHub). Submit all your code, but please exclude the **node_modules** folder.

Task 3: OPTIONAL Mash-Up Difficulty: Medium, but takes a lot of time.

If you have time to spare over the holidays, create a mash-up on a certain topic.

Wild Idea: Content Aggregation Search Engine for Music Artists

- Users enter the name of an artist.
- Page offers different opportunities to play music of this artist through different content providers, e.g. Spotify, Deezer, Rhapsody/Napster, or Microsoft Groove.
- Page shows YouTube Videos containing the name of the artist
- Page embeds tour dates of the artist
- Page displays news about the artist.

Your Mash-Up does not have to be centered around music artists / bands. It can also be something completely different. Be creative, we look forward to your mash-ups!

Submit all your code files and/or a link to a webpage where you show off your work.

Happy Holidays!

Enjoy the break! See you in 2017.

Sincerely yours,

The MMN Team

Submission

Please turn in your solution via UniWorX. You can form groups of up to three people.

We encourage you to sign up for Slack! All you need is a CIP account and an email address that ends in “@cip.ifi.lmu.de”. Ask us if you don’t know how to get them.

If you have questions or comments before the submission, please contact one of the tutors. They are on Slack @tobi.seitz, @peterjuras and @thomas-weber. Remember, that they also want to enjoy their weekends ☺

It also makes sense to ask the question in our #mmn-ws1617 channel. Maybe fellow students can help or benefit from the answers, too!

CodeLabs

As always, everyone is invited to participate in the CodeLabs on Wednesday between 6 and 8 pm. We encourage you to go there if you struggle with installation or nasty bugs that just don’t seem to go away. The times are flexible, so you can join anytime during the CodeLab hours.