Sovellusprojekti Kepler, 1. lähdekoodin katselmointi

Aika Torstai 9.4.2015 klo 10:15 – 11:38
Paikka Jyväskylän yliopisto, Agora, Sovellusprojektien kokoushuone C226.1

Läsnä
Projektiryhmä
Anu Koskela
Mikko Kuhno
Henrik Paananen
Atte Räty
Joonas Konki, sihteeri
Ohjaajat
Petri Partanen
Jukka-Pekka Santanen

Muistio

Muistio laadittu: 13.4.2015
Muokattu: —

Katselmoinnin yleiset huomiot

Katselmoinnissa käytiin läpi Partasen kommentteja Kepler-sovelluksen lähdekoodista ja toteutusratkaisuista. Partanen totesi ryhmän päässeen hyvin alkuun sovelluksen kehittämisessä ja lähdekoodin tyylin olevan pääosin hyvällä tasolla.

Kepler-nimeä ei kannata käyttää tiedostojen tai luokkien nimissä, koska se voi hankaloittaa myöhemmin sovelluksen käyttöä toisissa projekteissa.

Sivujen staattinen sisältö tulisi siirtää templates-hakemistosta oikeaan paikkaan. Kolmannen osapuolen JavaScript-kirjastoista kannattaa sovelluksen lopulliseen versioon lisätä mukaan vain tarpeelliset .min.js-päättelliset tiedostot.

Käyttöliittymän jQuery-kirjastolle voisi käyttää uudempaa versiota (2.X), ellei vanhemman (1.X) version käytölle ole perusteita.

Kaikki tarvittavat tiedostot ei ole listattuina MANIFEST.in-tiedostossa. Vaihtoehtoisesti tämän tiedoston voi poistaa ja käyttää ulkopuolista BSD-lisenssin alaista setuptools-git -kirjastoa, jonka voi lisätä setup.py-tiedostoon. Kaikkia tarvittavia ulkopuolisia komponentteja ei ole myöskään listattu setup.py-tiedostoon.

Partasen mukaan luokkien, funktioiden ja tiedostojen alkujen kommentointia tulisi lisätä kaikkialle lähdekoodiin. Tiedostojen versionumerointia ja päiväyksiä on käytettävä johdonmukaisesti yhdessä sovitulla tavalla. Copyright-tiedoissa voisi lukea 2015 Kepler project members.

Projektiryhmä kysyi, onko tietokannan ID-numeroiden lähettäminen palvelinpuolen sovelluksen ja käyttöliittymän välillä järkevää. Viestintä on salatun HTTPS-yhteyden yli, eikä tietokantaan pääse tekemään muutoksia muualta kuin sovellusta ajavalta palvelimelta käsin. Näin ollen oikeiden tietokannan ID-numeroiden käytön ei katsottu olevan ongelmallista. Tietoturvaasioihin liittyen ryhmän kannattaa käydä läpi esimerkiksi OWASP Top 10 -lista.

Palvelinpuolen toteutukseen liittyvät huomiot

Pelkästään muihin käyttöliittymän sivuille sisällytetyille osille ei ole tarpeen määritellä reittejä kepler/views.py tiedostossa. Esimerkiksi käyttäjän ei tarvitse avata sivujen yläosan top_bar-sivua suoraan sellaisenaan.

Jokaiseen Python-lähdekooditiedoston alkuun tulee lisätä ”-*- coding: utf-8 -*-”, jos halutaan sovelluksen tukevan vanhempaa Python 2-versiota oikein.

Kommentointia ja docstringejä lisättävä lähdekoodiin, myös tiedostojen alkuun. Koodirivien pituus saa olla maksimissaan 79 merkkiä.

Yleisesti Python-lähdekoodissa vakiomuuttujat tulisi mieluiten kirjoittaa kokonaan isoilla kirjaimilla (auth.py). Pitkien funktioiden nimiä kannattaa lyhentää, ja aihepiirin mukaan nimettyjä asioita nimetä yleisemmin. For-silmukoiden ja listojen muodostamisessa kannattaa käyttää Pythonin näppärää ”list comprehensions”-ominaisuutta.

Views.py-tiedoston import-lauseita tulisi miettiä tarkemmin (import pyramid.httpexceptions vai from pyramid.httpexceptions import HTTPTemporaryRedirect). Kovakoodattujen login-osoitteiden tilalle voi käyttää request.route_url(’login’), jos tarkoitus on käyttää uudelleenohjauksia. Muistettava varoa mahdollista login-silmukan muodostumista kehityksen aikana. Ylimmän tason funktioiden väliin tulee jättää kaksi rivinvaihtoa yhden sijaan. Eri näkymien oikeudet kannattaa nimetä tarkemmin ja selvästi käyttäjäroolien oikeuksista erottuvasti.

Try-lohkojen sisällä olevat koodit ovat välillä hieman liian pitkiä. Kaikkia poikkeuksia ei kannata yrittää hallita, vaan antaa Pyramidin ottaa kiinni suurin osa poikkeuksista. Tietokantakyselyissä kannattaa käyttää one() kutsun sijaan parempaa kutsua first(), joka palauttaa tarvittaessa arvon ”None”, eikä aiheuta poikkeusta, jos löytyy useampi kuin yksi hakutulos.

Wildcard-importtia (from X import *) ei kannata käyttää tarpeettomasti.

Lähdekoodin tyyliä ja virheitä kannattaa analysoida pylint-ohjelmalla.

Käyttöliittymän toteutukseen liittyvät huomiot

JavaScript-koodirivien pituus saa olla maksimissaan 79 merkkiä. JavaScriptille ominaista camelCase-nimeämiskäytäntöä kannattaa käyttää. Kommentoinnit on lisättävä (JSDoc, http://usejsdoc.org). Nimiavaruuksia kannattaa pienentää sekä käyttää vähemmän globaaleja muuttujia. Ihannetilanteessa koko JavaScript-tiedosto on siirretty anonyymin funtion sisään (function(){ ... }());.

Kepler.js-tiedostossa kannattaa palvelimen osoite hakea dynaamisesti (var host = window.location.origin;) kovakoodatun osoitteen sijaan. Kepler-luokan objektia ei välttämättä tarvitse luoda new-sanalla, vaan voisi sijoittaa sen suoraan muuttujaan. Yhtäsuuruuksissa kannattaa virhetilanteiden välttämiseksi käyttää kolmea yhtäsuuruusmerkki kahden sijaan.

JavaScript-lähdekoodin tyylin ja virheiden analysointiin kannattaa käyttää JSHint-ohjelmaa ja HTML-sivuille esimerkiksi sivun https://html5.validator.nu/ HTML5 Validator -ohjelmaa.

Partanen ehdotti, että olisi järkevää sisällyttää kaikki projektin oma toiminnallisuus yhteen JavaScript-tiedostoon. HTML, CSS ja JavaScript on pidettävä puhtaasti erillään toisistaan.

Käyttöliittymän klikkaustoiminnot voisi määritellä JavaScript-tiedostossa esimerkiksi tyyliin $('myButton').click(myButtonFunction); tai $('myButton').on('click', this.myButtonFunction);.