GitHub-ohjeita

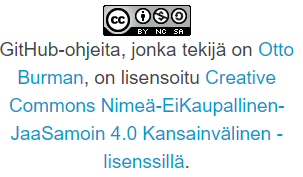
Kesä 2019, Otto Burman

Lähde: <https://guides.github.com/>

# ALKUTOIMET

Kirjaudu GitHub-sivustolle ja asenna GitHub Desktop-sovellus:

1. Käynnistä <https://github.com/> ja noudata ohjeita oman GitHub-tilin luomiseksi. Kun sinulla on olemassa käyttäjätunnukset niin GitHubiin pääset sisään linkistä <https://github.com/login>.
2. Asenna GitHub Desktop omalle laitteelle. Windows ympäristössä toimiva GitHub Desktop saadaan ladattua osoitteesta <https://desktop.github.com/>.
3. Asenna myös XAMPP, jotta kykenet seuraamaan viimeisetkin esimerkit.



Sisällys

[ALKUTOIMET 1](#_Toc13814912)

[1 Johdantoa GitHubin käsitteisiin 3](#_Toc13814913)

[1.1 Repository-arkisto 4](#_Toc13814914)

[1.2 Branching-haarautuminen 5](#_Toc13814915)

[1.3 Commit changes-muutosmerkinnät 6](#_Toc13814916)

[1.4 Pull-Request-pyynnöt 7](#_Toc13814917)

[1.5 Merge pull request-muutoksien käyttöönotto 8](#_Toc13814918)

[2 EkaProjekti 9](#_Toc13814919)

[2.1 Repository-arkisto 10](#_Toc13814920)

[Arkiston luonti 10](#_Toc13814921)

[2.2 Branch-haarautuma 12](#_Toc13814922)

[Tekemisen vaiheet 12](#_Toc13814923)

[2.3 Commit Changes-muutoksien kirjaaminen 13](#_Toc13814924)

[2.4 Pull Request-muutospyyntö 15](#_Toc13814925)

[2.5 Merge-muutoksien liittäminen päähaaraan 17](#_Toc13814926)

[2.6 Download Repository-vie tiedot omalle koneelle 18](#_Toc13814927)

[2.7 GitHub Desktop-käytön alkeet 19](#_Toc13814928)

[2.8 Tarpeettoman GitHubin Repository-arkiston poistaminen 21](#_Toc13814929)

[3 PHP-projekti 22](#_Toc13814930)

[3.1 Repo-arkiston luonti ja käyttö 22](#_Toc13814931)

[3.2 Branch-haarautumien hallinnointia 24](#_Toc13814932)

[3.3 Tiedoston editointi 24](#_Toc13814933)

[3.4 Commits-muutosmerkinnät 25](#_Toc13814934)

[3.5 Pull Request-pyynnöt 27](#_Toc13814935)

[Notifikaatioluettelo 27](#_Toc13814936)

[Open pull Request 27](#_Toc13814937)

[3.6 Master Branch päivitys 28](#_Toc13814938)

[Sähköpostin viestit eli GitHub-notifikaatiot 29](#_Toc13814939)

[3.7 Muutokset tuotantoympäristöön 30](#_Toc13814940)

[Open in Desktop-kloonaus 31](#_Toc13814941)

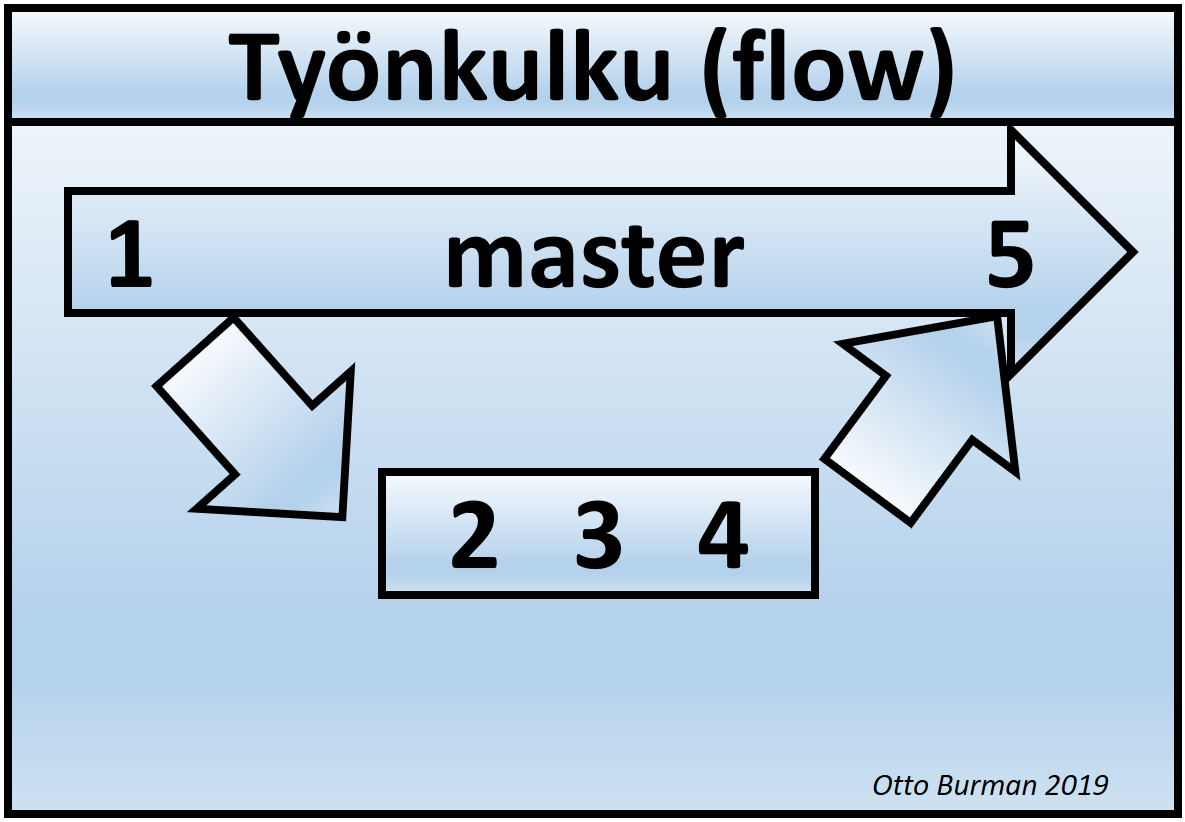
[Download ZIP-kopiointi 34](#_Toc13814942)

[4 Hankittu ja tavoiteltu osaaminen 36](#_Toc13814943)

[Tekijän yhteystiedot 37](#_Toc13814944)

# Johdantoa GitHubin käsitteisiin

Oheinen kuvio liittyy peruskäsitteisiin ja työn kulkuun GitHub-projekteissa. Numerot viittaavat sivun alalaidassa olevaan luetteloon.



GitHub-flow, lähde: <https://guides.github.com/introduction/flow/>

GitHub soveltuu sovelluskehityksen työnkulkua kuvaavan tapahtumavirran (flow) dokumentointiin. Työnkulun dokumentointi alkaa arkiston (repository) perustamisella. Työnkulun tapahtumavirrassa on erilaisia työnkulun päähaarasta (master-branch) eriytyviä haarautumia (branches). Työnkulku kuvaa tiimien ja projektien jatkuvaa etenemistä kohti uutta paranneltua tuotosta. Kehitysprojektin kohteena oleva tuotosta tai sovellusta kehitetään tai ylläpidetään, samalla kun uusien versioiden käyttöönotto tapahtuu hallitusti ja tarkoituksenmukaisesti dokumentoituna.

Tämän ohjeen ensimmäiset luvut kuvaavat peruskäsitteet sekä pyrkii selvittämään, miten ja miksi GitHubissa työnkulkua dokumentoidaan. GitHubin tarjoamia välineitä voidaan käyttää internetissä GitHub-sovelluksella (<https://github.com/> ) ja/tai kehitystyötä voi tehdä myös paikallisesti omalla tietokoneella GitHub Desktop-työpöytäsovelluksella esimerkiksi Windows ympäristössä (<https://desktop.github.com/>). Sovellukset vuorovaikuttavat keskenään ja mahdollistavat monipuolisen sovellusten kehitysympäristön: tietoja voidaan helposti siirtää internet-palvelimelle ja takaisin omalle laitteelle. Aluksi keskitytään GitHubin internetversion viiteen peruskäsitteeseen kuten

1. Repository-arkisto
2. Branching-haarautuminen
3. Commit changes-muutosmerkinnät
4. Pull Request-pyynnöt
5. Merge pull request-muutoksien käyttöönotto

Kukin käsite käydään läpi seuraavissa luvuissa, joiden ymmärtämiseksi on asennettava GitHub-käyttäjätili: Käynnistä <https://github.com/> ja noudata ohjeita. Kun sinulla on olemassa käyttäjätunnukset niin GitHubiin pääset sisään selaimella linkistä <https://github.com/login>.

## Repository-arkisto

GitHubin tapahtumavirran (flow) dokumentointi alkaa Repository-arkiston perustamisella.

Selaimen komento <https://github.com/login> avaa aloitusnäkymän. Aloitusnäkymään pääsee aina myös selainnäkymän vasemmassa yläkulmassa olevan koti-symbolin avulla:



Sivun vasemmassa yläosassa olevalla vihreällä New-painikkeella luodaan uusi Repository-arkisto:

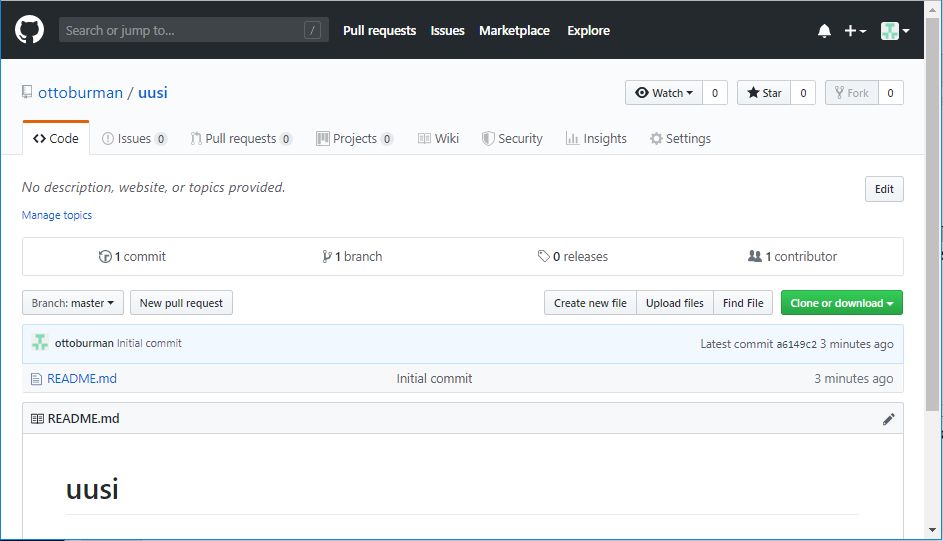


Avautuvassa uudessa näkymässä Repository-arkisto nimetään, lisätään lyhyt kuvaus, luodaan arkistoa varten README-tiedosto laittamalla rasti ao. ruutuun sekä määritetään mahdollisesti arkiston käytölle lisenssiehtoja.

Lopuksi painetaan vielä varmistukseksi vihreää Create repository-painiketta:



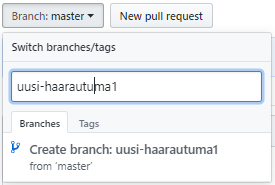
Painike avaa näkymän uuteen Repository-arkistoon (ottoburman/uusi):



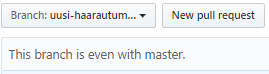
## Branching-haarautuminen

Kehitysprojekteihin liittyy uusien ideoiden ja ajatusten kokeileminen. Jotkut ideoista saattavat olla varsin valmiita, mutta osa taas vaatii edelleen kehittämistä. Projektin edetessä sen työnkulku usein haarautuu alaprojekteihin, joissa uusia ajatuksia kehitellään omina työvaiheina. GitHubissa käsite Branch-haarautuminen auttaa hallitsemaan tätä sovelluskehitykseen liittyvää varsin tyypillistä etenemistä ja kehitysvaihetta.

GitHubin työnkulun kuvaukseen liittyy kaksi haarautumaa: master-branch ja erikseen nimettävä kehitystyön branch-haara. Kaikissa valmiissa, tuotannossa olevissa, sovelluksissa on aina master-branch-haara, jonka tiedot tai tiedostot ovat valmiin tuotantoversion tietoja. Kun tuotantoversiota ryhdytään kehittelemään edelleen, niin tällöin on tarkoituksenmukaista luoda työskentelyä varten uusi arkisto tai työtila, joka eristetään tuotantoversion tiedoista. GitHubissa tämä työtila nimetään omaksi uudeksi branch-haarautumaksi. Repository-arkiston ikkunassa Branch:master-painike avaa ikkunan, johon voidaan antaa uudelle haarautumalle nimi ja luoda se Create branch:-valinnalla alla olevan ohjeen mukaisesti.



Uuden haaran luomisen yhteydessä syntyy siis kehitysympäristö, jossa voidaan turvallisesti kokeilla ideoita. Luodussa uusi-haarautuma1-kehitysympärisössä tehdyt muutokset eivät vaikuta master-branch-päähaaran tietoihin. Master-branch-haaraan tehtävät muutokset täytyy hyväksyä GitHub-järjestelmän rutiinien avulla, joita käsitellään hetken kuluttua. Tässä vaiheessa on oleellista huomata, että Repository-arkistossa näkymä on nyt tähän uuteen nimettyyn ”uusi-haarautuma1”-työskentelyalueeseen. Tämä nimi näkyy esimerkiksi Branch-painikkeen otsikossa, jonka alapuolella on huomautus, että haarauma on samanlainen sen päähaaran kanssa (even with master).



Haarautuminen, branching, on GitHub-flow-työvirran hallinnan keskeinen käsite, ja koko GitHub-työskentely perustuu käsitteen ymmärtämiselle. Tärkeää on ymmärtää, että kaikki master-branch-haarassa olevat ominaisuudet ovat eritetty kehityskokeiluista aina alkuperäisen sovelluksen käytössä ja uudelleen asennettavissa tuotantoympäristöön.

Tärkeää sovelluskehityksessä on, että uusi haara erotetaan master-haarasta, aina kun työskennellään uuden ominaisuuden tai sovellukseen liittyvien kriittisten toimintojen parissa. Uuden haaran nimen tulisi olla mahdollisimman kuvailevaa kehitystyön tavoitteiden kannalta (esim. käyttäjän-autentikointi, tietokantayhteys, käyttäjän poistaminen), jotta muut voivat nähdä, minkä parissa tätä kehitystyötä tässä haarassa juuri nyt tehdään.

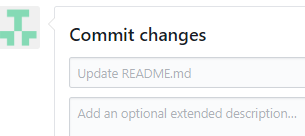
Huomaa, että ennen haarauman tekoa on luotava Repository-arkisto, johon voidaan siirtyä suoraan pääsivulta. Muutoksien tekeminen aloitetaan vasta kun ollaan varmasti siirrytty tähän uuteen haarautumaan. Tämä tulee varmistaa Branch-painikkeen otsikosta (Branch: uusi-haarautuma1).

## Commit changes-muutosmerkinnät

Tässä johdannossa tehdään nyt muutos uusi-haarautuma1-arkiston README.md-tiedostoon (varmista, että ollaan oikeassa työtilassa eli Branch: uusi-haarautuma1). README.md-tiedoston muutos tehdään painamalla editointi-painiketta

,

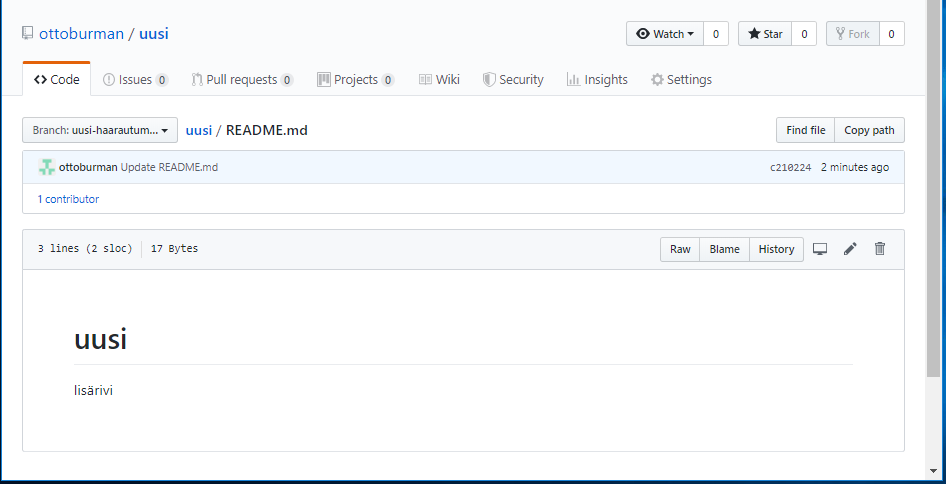
jolloin voidaan kirjoittaa esimerkiksi lisää rivejä tiedostoon. Tehdyt muutokset dokumentoidaan sivun alalaidassa olevaan tietolaatikkoon:



Lopuksi tehdyt muutokset vahvistetaan **Commit changes**-painikkeella (tai peruutetaan Cancel-näppäimellä):



Commit-verbille voidaan käyttää suomennosta kommitoida (sanakirja.org). Englanniksi verbistä on käytetty esimerkkiä: “The act of committing (e.g. a database transaction or source code into a source control repository), making it a permanent change”. Tietojenkäsittelyssä kommitoinnilla voidaan siis tarkoittaa pysyvän muutoksen aikaansaamista. GitHubissa toteutettuja ja tallennettuja muutoksia seurataan commit-merkinnöillä, joihin liittyy commit-viesti, joka on lyhyt kertomus siitä miksi muutos on tehty. Commit-viestit tallentuvat GitHubin muutoshistoriaan, joka auttaa muita ymmärtämään, mitä muutoksia sovelluksen parantamiseksi on tehty.



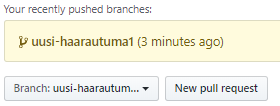
Aina kun haaraan lisätään, muokataan tai poistetaan tietoja, on nämä muutokset kirjattava. Tehty muutos dokumentoidaan tekemällä commit-merkintä. Arkistossa commit-lisäysprosessi (adding commits) pitää kirjaa työn edistymisestä. Commit-merkinnät luovat kaikille avoimen näkymän muutostyöstä: muut voivat seurata ja ymmärtää mitä muutoksia on tehty ja miksi. Jokaiseen commit-merkintään tulee siis liittää kuvaus, miksi tietty muutos tehtiin. Lisäksi jokainen commit-merkintä katsotaan omaksi erilliseksi muutokseksi. Tällä menettelyllä voidaan palata aikaisempiin tilanteisiin: muutoksia voidaan korjata.

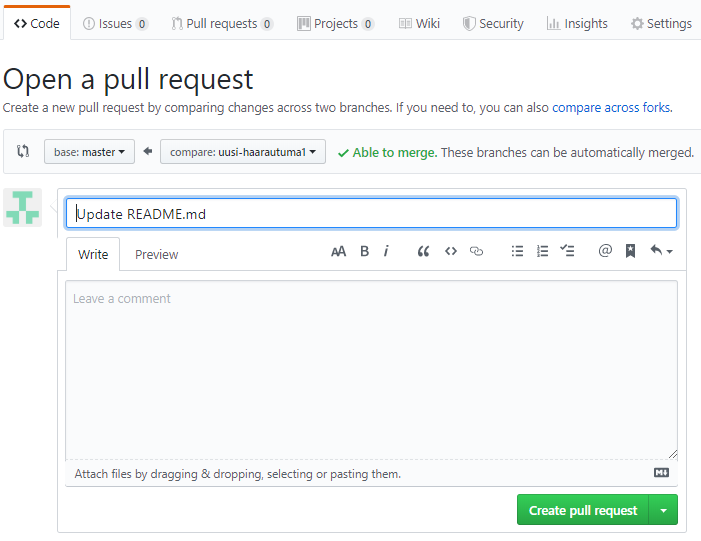
Commit-merkinnät ovat kehitystyössä erittäin tärkeitä. GitHub seuraa automaattisesti muutoksia ja näyttää ne sitten commit-merkintöinä, kun esimerkiksi muutokset ovat toteutettu tuotantoversio on päivitetty. Kirjoittamalla selkeitä merkintöjä, helpotetaan muiden mahdollisuutta seurata kehitystyötä ja antaa palautetta. Palataan Repository-arkistoon esimerkiksi painamalla vasemmalla ylhäällä olevaa Code-lipuketta

, ja siirrytään seuraavaan vaiheeseen…

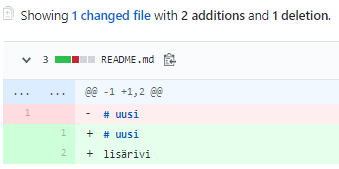
## Pull-Request-pyynnöt

Repository-arkistossa painetaan New pull request-näppäintä

, joka avaa keskustelun muutoksista perustuen tehtyihin commit-merkintöihin. Merkinnät ovat integroituneet GitHub-arkistoon ja kuka tahansa voi nähdä, mitä muutoksia liitetään sovellukseen, kun pyyntö hyväksytään. Painalluksen jälkeen avautuu ikkuna, johon uudet commit-merkinnät tehdään / lisätään. Merkinnät kuitataan valmiiksi painamalla lopuksi Create pull request-painiketta.



Open a pull request-näkymän alaosassa ovat tämä esimerkin muutokset: lisätty rivi ja teksti.

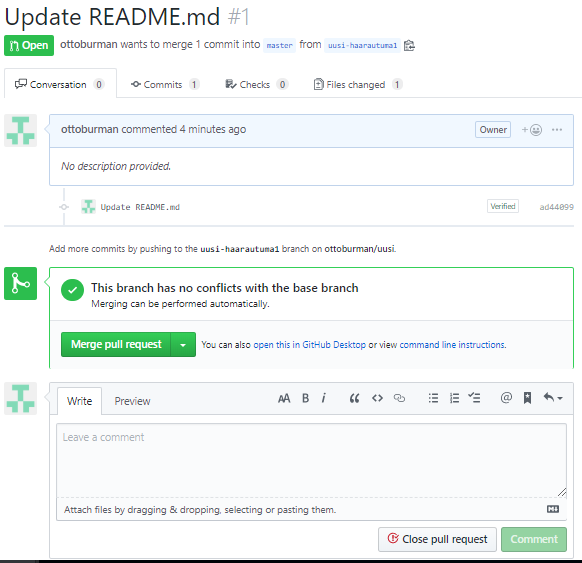


Pull-Request-pyyntö voidaan avata milloin tahansa kehitysprosessin aikana. Esimerkiksi heti kehittelyn alussa ja haluat jakaa joitakin kuvakaappauksia tai yleisiä ideoita. Käyttämällä GitHubin asetuksia Pull Request-viestit lähetetään nimettyihin sähköpostiosoitteisiin, jolloin palautetta voi pyytää/ saada nimetyiltä henkilöiltä (Valitse Settings-lipuke ja Notifications-osio, johon merkitään s-postilistat).

Pull-Request-pyynnöt käynnistävät koodin tarkistuksen ja keskustelun ehdotetuista muutoksista ennen kuin muutokset yhdistetään master-päähaaraan. Työntekoa voi jatkaa aina uudessa haarautuman alahaarassa saadun palautteen perusteella. Muutokset voidaan palauttaa takaisin lähtötilanteeseen. Pull Request-kommentit on kirjoitettu Markdown-koodikielellä, joten tekstiin voi upottaa kuvia ja emoji-tiedostoja, käyttää valmiiksi muotoiltuja tekstilohkoja ja muita tekstiä keventäviä muotoiluja.

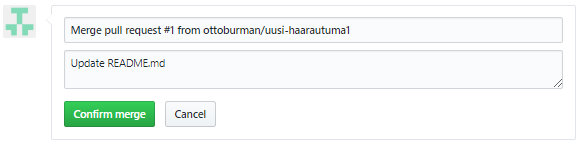
## Merge pull request-muutoksien käyttöönotto

Edellisen sivun Create pull request-painike avaa Update README.md-näkymän:



Kun Pull-Request-pyyntö/pyynnöt on tarkistettu ja uusi-haarautuma1 on hyväksytty niin, muutokset voidaan vielä testata esimerkiksi todellisessa tuotantoympäristössä. Jos uusi versio aiheuttaakin ongelmia, voidaan tilanne palauttaa ottamalla käyttöön nykyinen master-versio. Kun tuotantoympäristössä tehdyt testaukset osoittavat kaiken olevan kunnossa, voidaan muutokset (esim. uusi sovelluksen koodi) yhdistää (merge) master-haaraan. Ensin painetaan Merge pull request-näppäintä:

, joka avaa näkymän, jossa Confirm merge-näppäintä painamalla vahvistetaan tietojen yhdistäminen:



Kun koodi yhdistetään, niin Pull-Request-pyynnöt tallentavat koodin historian ja tehdyt muutokset. Muutokset ovat jäljitettävissä ja ne antavat mahdollisuuden ymmärtää miksi ja miten päätös muutoksista tehtiin. Kehityshaara on nyt tarpeeton ja se voidaan poistaa Delete branch-painikkeella:



# EkaProjekti

GitHub on ohjelmakoodin hallintaan (hosting) tarkoitettu alusta, jonka avulla toteutetaan sovelluskehityksen versiohallintaa ja yhteisöllistä työskentelyä kehitysprojektien parissa. Projektityöskentelyn voi väittää olevan paikasta ja ajasta riippumatonta. Tässä luvussa tutustutaan GitHubin tärkeimpiin projektityöskentelyyn liittyviin toimintoihin. Näitä ovat Repository-arkistojen luominen, Branch-haarautumien tekeminen, Commit-muutosmerkintöjen kirjoittaminen ja Pull Request-pyyntöjen toteuttaminen. Esimerkin avulla tehdään oma EkaProjekti-Repository-arkisto ja opitaan GitHubin työnkulku yhteisen tiedon luomiseksi ja sen tarkastelemiseksi sekä edelleen kehittämiseksi. Tämän esimerkin läpikäyntiin tarvitaan GitHub.com-tili ja internet-yhteys.

Aloittaaksesi esimerkin läpikäynti käynnistä <https://github.com/> ja noudata ohjeita oman tilin luomiseksi. Jos sinulla on jo olemassa käyttäjätunnukset, niin pääset tiliisi linkistä <https://github.com/login>. Tässä vaiheessa ei vielä edellytetä ohjelmointiosaamista, komentorivikomentojen käyttämistä tai varsinaisen Git-ohjelmiston asentamista, jonka päälle versionhallintaohjelmisto GitHub on rakennettu.

Tämä ohje kannattaa avata toiseen selainikkunaan, samalla kun luet ohjeen vaiheita. Tavoitteena on tässä luvussa oppia miten

1. Luodaan työskentelytila ja käytetään Repository-arkistoa
2. Aloitetaan uusi työtila ja hallinnoidaan uutta Branch-haaraa
3. Tehdään muutoksia tiedostoon ja työnnetään (Push) ne GitHub-palveluun Commits-merkintöinä
4. Avataan Pull Request-pyyntö
5. Päivitetään tuotantoversio-merge Branch: master

## Repository-arkisto

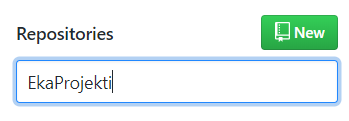
Repository-arkistoa käytetään tavallisesti yksittäisen kehitysprojektin hallintaan. Arkistot voivat sisältää kansioita ja tiedostoja, kuvia, videoita, laskentataulukoita ja tietojoukkoja – siis mitä tahansa kehitysprojektiin liittyvää tietoa. Tärkeää on aina luoda projektille README.md-tiedosto, johon kirjataan perustietoja projektista. GitHubin avulla tiedosto syntyy samalla kun luot arkiston, kunhan muistaa laittaa rastin ao. ruutuun. Arkiston luonnin yhteydessä on mahdollista liittää mukaan myös lisenssiehtoja sisältävän tiedoston. Repository-arkistoon tallennetaan kehitysideoita, muistioita sekä ohjelmakoodia, joita voi jakaa muille ja saada palautetta omista kehittämisajatuksista.

### Arkiston luonti

GitHub-sivuston vasemmassa yläkulmassa on käyttäjän koti-ikkunaan siirtävä symboli:

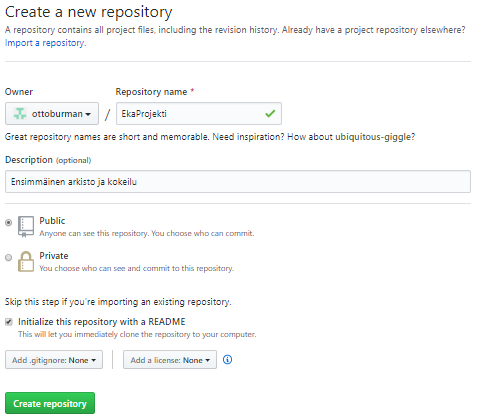


Napsauttamalla GitHub-symbolia päästään aina aloitusnäkymään, jossa luodaan tai otetaan käyttöön jo aikaisemmin luotuja Repository-arkistoja. Näkymä näyttää tältä ensimmäisellä kerralla:



Repositories-laatikkoon kirjoitetaan EkaProjekti, mallin mukaisesti. Huomaa, että jos projektin arkisto on jo olemassa, se näkyy tässä ikkunassa. Kokonaan uuden arkiston luomiseksi painetaan vihreää New-näppäintä. Uusi ikkuna avautuu. Täytä tiedot alla esitetyn mallin mukaisesti. Tärkeät kolme kohtaa ovat

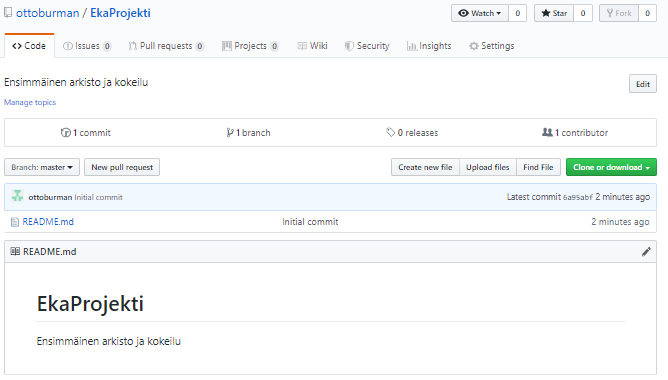
1. arkiston nimeksi laitetaan EkaProjekti.
2. kirjoitetaan lyhyt kuvaus: ”Ensimmäinen… ”
3. valitaan kohta Initialize this repository with README.



Kehitysprojektille on järkevää antaa sille kuvaava nimi ja arkistoon luoda liittyvä README.md-tiedosto, johon kirjataan keskeistä tietoa kehitysprojektista. Tässä vaiheessa voidaan käydä tutustumassa eri lisenssiehtoihin valitsemalla **Add a License**-painikkeen vaihtoehdoista.

Lopuksi painetaan vihreää Create repository-näppäintä, joka avaa näkymän luotuun arkistoon.

Näkymä ottoburman/EkaProjekti-Repository-arkistoon:



Näkymässä voidaan korjailla ja täydentää kuvauksia (Edit-painike tai kynä-symboli ), luoda (Create new file) uusia ja lisätä (Upload files) sekä hakea (Find file) olemassa olevia tiedostoja. Esimerkiksi Upload files-painike avaa ikkunan, jonka avulla voi kopioida omalta kovalevyltä kansioita ja tiedostoja GitHubin Repository-arkistoon. Tiedostojen hallintaan tarkoitettuja komentopainikkeita ovat seuraavat:



Repository-arkisto voidaan myös kopioida sellaisenaan GitHubDesktop-sovelluksella käytettäväksi paikallisella tietokoneella **Clone or download**-painikkeen avulla. Tähän palataan myöhemmin.

Repository-arkisto on tarkoitettu tavallisesti yhden yksittäisen kehitysprojektin hallintaan. Repository-arkisto voi sisältää useita kansioita ja näissä tiedostoja kuten kuvia, videoita, laskentataulukoita ja muita tietomuotoja. Arkistossa voi olla siis mitä tahansa projektiin liittyvää tietoa.

Tärkeää on nyt tässä vaiheessa huomata kaksi painiketta: Branch- ja New pull request-painikkeet:



Branch-painikkeessa näkyy, että olemme tällä hetkellä päähaarassa eli näppäimessä on merkintä Branch: master. Seuraavaksi luodaan uusi työskentelytila eli haarautuma. Haarautuma on siis kopio nyt luodusta master-päähaarasta.

## Branch-haarautuma

Brancing-haarautuminen on tapa käsitellä Repository-arkiston eri versioita. Oletuksena arkistossa on yksi pääversio, jonka nimi on master, jota myös pidetään projektin tuotantoversiona. Branch-haaroja käytetään kokeiluun ja uusien versioiden muokkaamiseen ennen kuin ne päivitetään (merge) pääversioon.

Kun luodaan uusi haarautuma päähaarasta (Branch: master), syntyy siis kopio tai oikeastaan tilannekuva (snapshot), juuri sellaisena kun se luotaessa oli. Jos joku toinen tekee samanaikaisesti muutoksia master-branch-haaraan, nämä päivitykset saadaan otettua käyttöön (pull in).

Nyt meillä on kaksi kopiota työskentelyalueista: Branch: master ja uusi Branch-haara, joka nimetään kehitystavoitteiden mukaisesti. Uusi Branch-haara tarkoittaa, että tiedostoista syntyy uusia versioita Repository-arkistoon.

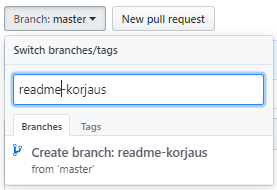
GitHubissa toimivat kehittäjät, kirjoittajat ja suunnittelijat käyttävät haaroja korjausten säilyttämiseksi erillään Master-Branch-haarasta (tuotantoversiosta). Kun muutos on hyväksytty ja valmis, kehittäjät yhdistävät luomansa versiot omista Branch-haaroista Master-Branch-haaraan eli päivittävät tuotantoversiota.

### Tekemisen vaiheet

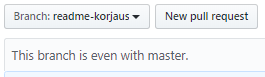
Varmistetaan aina, että ollaan oikeassa Repository-arkistossa. Sijainti näkyy sivun vasemmasta ylälaidasta, jossa on merkintä käyttäjätunnus/ Repository-arkisto, esimerkiksi



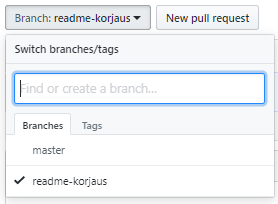
Napsauta Branch: master-painiketta. Tähän avautuvaan ikkunaan kirjoitetaan uuden haarautuman nimi (readme-korjaus) ja painetaan lopuksi Branches-lipukkeen alla olevaa Create branch: readme-korjaus-teksiriviä!



Ikkuna päivittyy ja muutos näkymässä varmistaa uuden haaran olevan olemassa ja sen olevan yhtäpitävä master-haaran kanssa:



Nyt käytössä on kaksi työskentelyaluetta ja työvirran haaraa, jotka ovat master ja readme-korjaus. Painamalla Branch: -näppäintä saadaan näkyviin tämän EkaProjekti-Repository-arkiston kaikki työskentelypolut eli Branch-haarat.



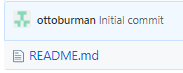
Seuraavaksi tehdään muutoksia uuteen readme-korjaus-Branch-haaraan.

## Commit Changes-muutoksien kirjaaminen

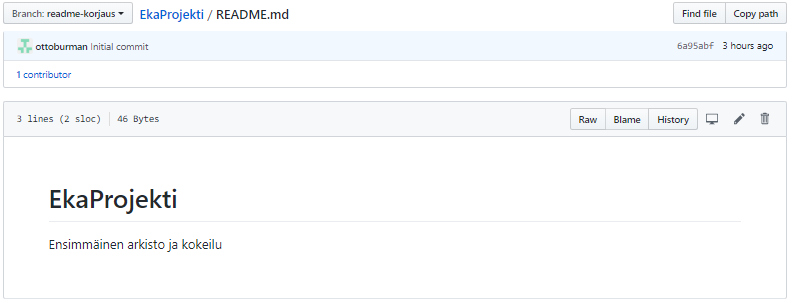
Tarkistetaan jälleen, että olemme varmasti oikeassa työalueessa eli readme-korjaus-haarassa, joka on siis kopio master-branch-haarasta. Tämä nähdään Branch-painikkeesta, jossa tulee olla teksti Branch: readme-korjaus, kuten edellisessä kuvassa oli tilanne. Haarautuman voi vaihtaa aina kuvakkeen alavalikosta ja siirtyä työskentelemään tälle haluamalleen työalueelle.

Tarkoituksena on tehdä joitakin muutoksia readme-korjaus-haarautuman tietoihin. GitHubissa toteutettuja ja tallennettuja muutoksia kutsutaan commit-sitoumuksiksi. Jokaisella muutoksella eli commit-sitoumuksella on siihen liittyvä kuvaus eli sitoumussanoma, joka on kertomus siitä miksi tietty muutos on tehty tai tehdään. Commit-sanomat tallentuvat muutoshistoriaan, joka on dokumentti sovelluksen kehitysvaiheista.

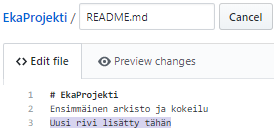
Korjataan tiedoston sisältöä napsauttamalle README.md-tiedoston nimeä:



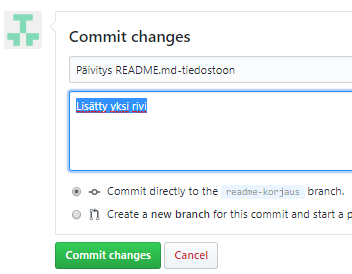
Nyt avautuu uusi editointi-näkymä tiedoston sisältöön.



Tiedostonäkymän oikeassa yläkulmassa olevaa kynä-kuvaketta napsautetaan, jonka jälkeen tieoja voidaan muokata.



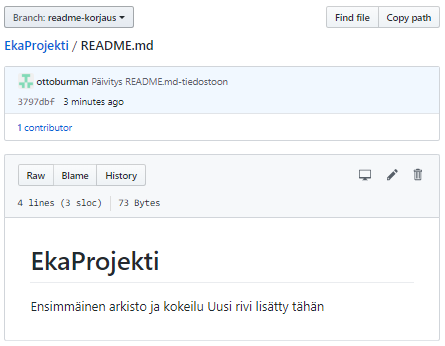
Avautuvassa editorissa lisätään uusi rivi: ”Uusi rivi lisätty tähän”.



Kirjoitetaan vielä viestille otsikko ja kuvaus siitä mitä on nyt tehty (”Lisätty yksi rivi”). Lopuksi napsautetaan Commit changes-painiketta:



Seuraavaksi avautuu jälleen uusi näkymä:

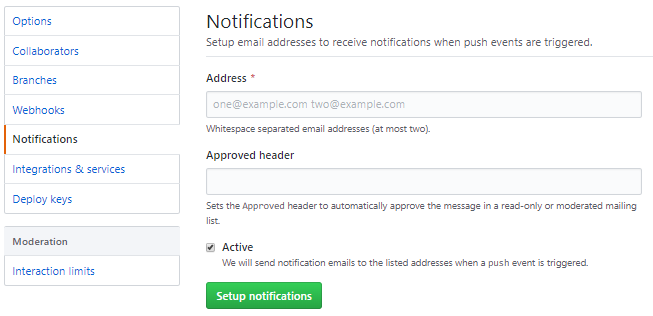


Näkymä näyttää readme-korjaus-haarautuman tilanteen ja tässä arkistossa sijaitsevan README.md-tiedoston sisällön. Haarautumassa on nyt uutta muuttunutta sisältöä, joka on erilaista kuin master-Branch-haarassa olevat tiedot.

Heti kun Commit-sitoumusmerkintä on tehty, voidaan Pull request-pyyntö avata ja aloittaa keskustelu jopa ennen kuin mitään varsinaisesti uutta koodia on olemassa. Käyttämällä GitHub-järjestelmän asetuksia voidaan palautetta pyytää joko tietyiltä ihmisiltä tai vaikka kokonaisilta tiimeiltä:

Valitse Settings-asetukset-lipuke

 ja näkymän vasemmasta laidasta avataan Notifications-asetukset, jolloin vastaanottajien osoitteet voidaan merkitä ao. kenttään:



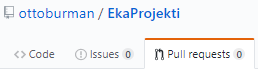
Lopuksi on muistettava painaa Setup notifications-painiketta.

Tämän oppaan esimerkissä Pull request-pyynnöt avataan käyttäjän omassa Repository-arkistossa ja muutokset hyväksytään ja yhdistetään omiin alkuperäisiin tietoihin. Tämä on yksi helppo tapa oppia ja harjoitella GitHub-työvirtaa ennen suurempien projektien toteuttamista.

## Pull Request-muutospyyntö

Lähettämällä Pull Request-pyyntö voidaan kertoa muille tehdyistä muutoksista. Kun Pull Request-pyyntö on lähetetty, sen vastaanottaneet osapuolet voivat tarkastella tehtyjä muutoksia ja voidaan alkaa keskustella mahdollisista parannusehdotuksista.

Pull request-pyynnön voi laatia esimerkiksi valitsemalla **Pull requests**-lipukkeen näkymän yläosasta:



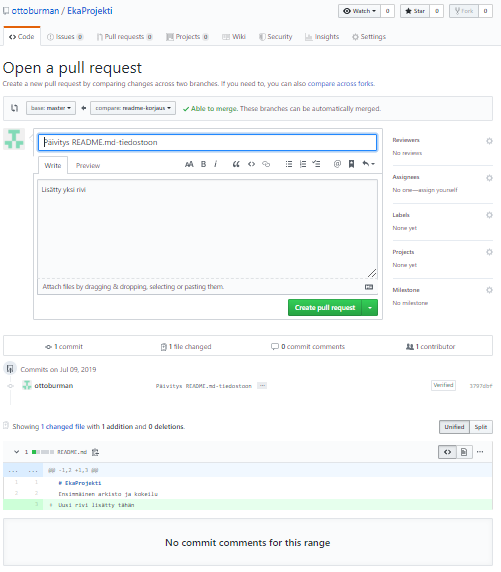
Valinta avaa uuden näkymän, josta voidaan valita seuraava toiminta joko Compare & pull request- tai New pull request-painikkeella:





Molemmat painikkeet vievät lopulta samaan tulokseen. Pedagogisista syistä valitaan nyt ylimmäinen **Compare & pull request**-painike, joka avaa automaattisesti näkymän, jossa verrataan haarautumien eroja ja tehtyjä muutoksia. (Huomaa, että klikkaamalla hiiren oikealla näppäimellä voidaan valita selaimen näkymän avautumisen uuteen ikkunaan, jolloin näiden kahden toimintojen vertailu on helpompaa.)

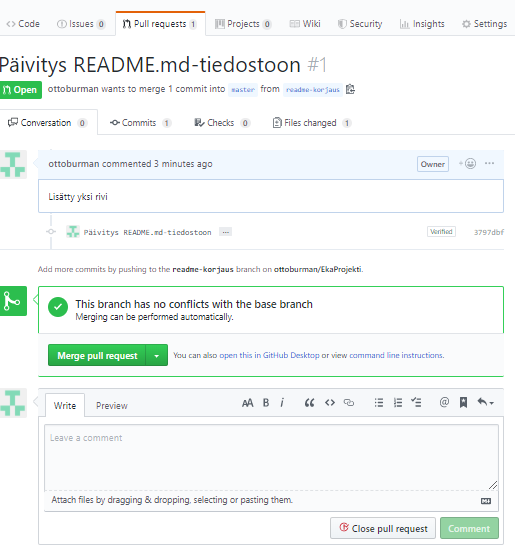
Pull request-pyynnöt ovat koko GitHubin yhteistyön ydintoimintaa. Pull request-pyyntö mahdollistaa erilaisia toimintoja kuten esimerkiksi muutoksien ehdottamiset ja pyyntöjen lähettämiset, jotta joku tarkistaisi nämä ja ottaisi käyttöönsä (vetää, pull) tekemäsi kehitysehdotukset yhdistämällä ne esimerkiksi omaan työskentelytilaansa. Compare & pull request-painike avaa seuraavan näkymän:



Pull request-pyyntö näyttää muutokset, tai oikeastaan siis sen, että sisällöt eroavat toisistaan. Muutokset, lisäykset ja poistot näkyvät vihreinä ja punaisina väreinä tietoja katseltaessa.

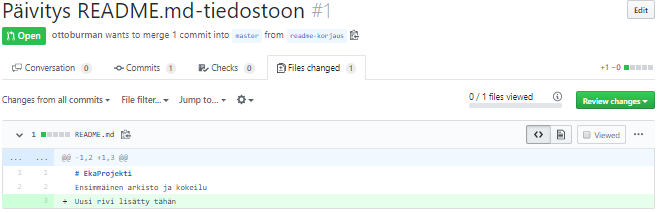
Näkymän alaosassa on raportti muutoksista: +-merkki osoittaa, että master-haaran readme.md tiedostoon ehdotetaan uutta tietoa: ”Uusi rivi lisätty tähän”. Tässä kohdassa varmistetaan, että muutokset ovat sellaisia kuin halutaan niiden olevan! Lopuksi painetaan vielä kerran vihreää Create pull request-painiketta.

Painalluksen jälkeen avautuu seuraava näkymä aloitetuista keskusteluista (Pull requests-välilehden Conversation-lipuke valittu):

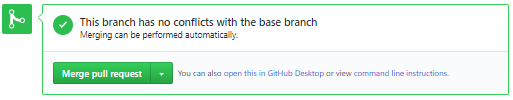


## Merge-muutoksien liittäminen päähaaraan

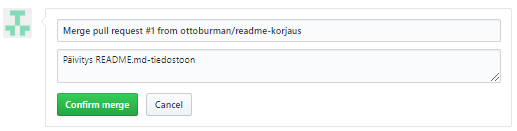
Pull requests-näkymässä voidaan vielä tarkistaa **Files changed**-lipukkeen alta mitä muutoksia olikaan tehty:



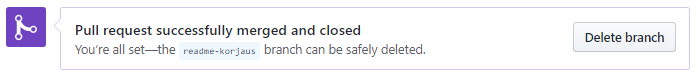
Koska muutokset ovat tarkoituksenmukaisia niin ne voidaan siirtää tuotantoversioon Merge pull request-painikkeella (**Conversation**-lipukkeen alla).



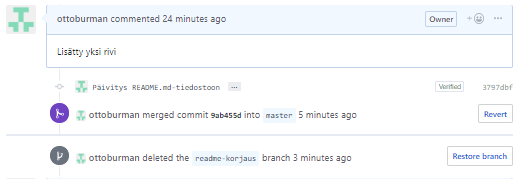
Tämä tulee vielä vahvistaa uudessa näkymässä:



Käytetty ja tarpeettomaksi käynyt Branch: readme-korjaus-haara poistetaan painamalla Delete branch-painiketta.



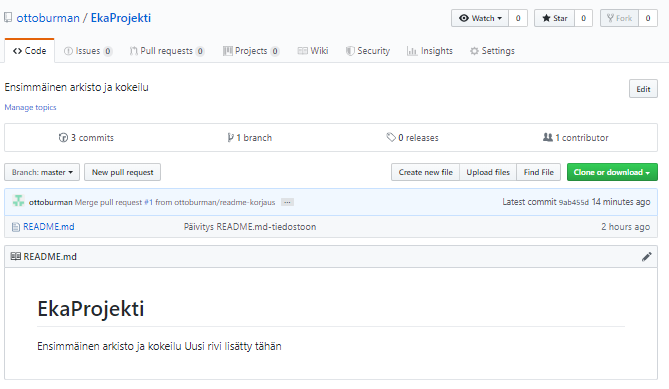
Työn kulku (work flow) on nyt dokumentoitunut näkymäksi:



Lopuksi voidaan siirtyä takaisin GitHubin ylätasolle esimerkiksi klikkaamalla

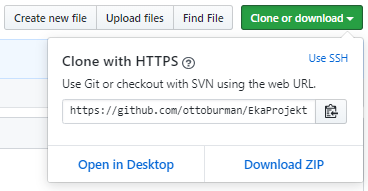


Täällä sivulla näkyvät tähän mennessä luodut Repository-arkistot: esimerkiksi äsken luotu EkaProjekti-arkisto. Klikkaamalla Repository-arkistoa siirrytään internetsivulle, jonka URL on [https://github.com/ käyttäjätunnus /EkaProjekti](https://github.com/%20käyttäjätunnus%20/EkaProjekti). Näkymä on seuraavassa kuvassa.

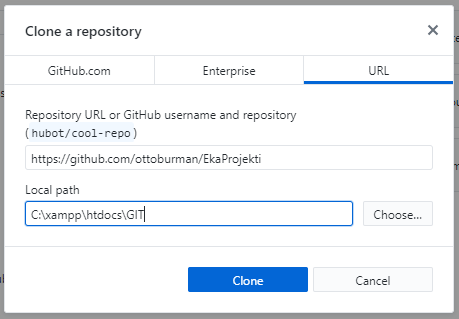


## Download Repository-vie tiedot omalle koneelle

Painamalla **Clone or download**-painiketta avautuu ikkuna, joka kertoo tietojen sijainnin URL-osoitteen (<https://github.com/käyttäjätunnus/>EkaProjekti). Tiedot voidaan ladata zip-tiedostona (Download ZIP-valinta) oletuskansioon tai arkiston tiedot voidaan sijoittaa haluttuun kansioon (Open in Desktop-valinta).



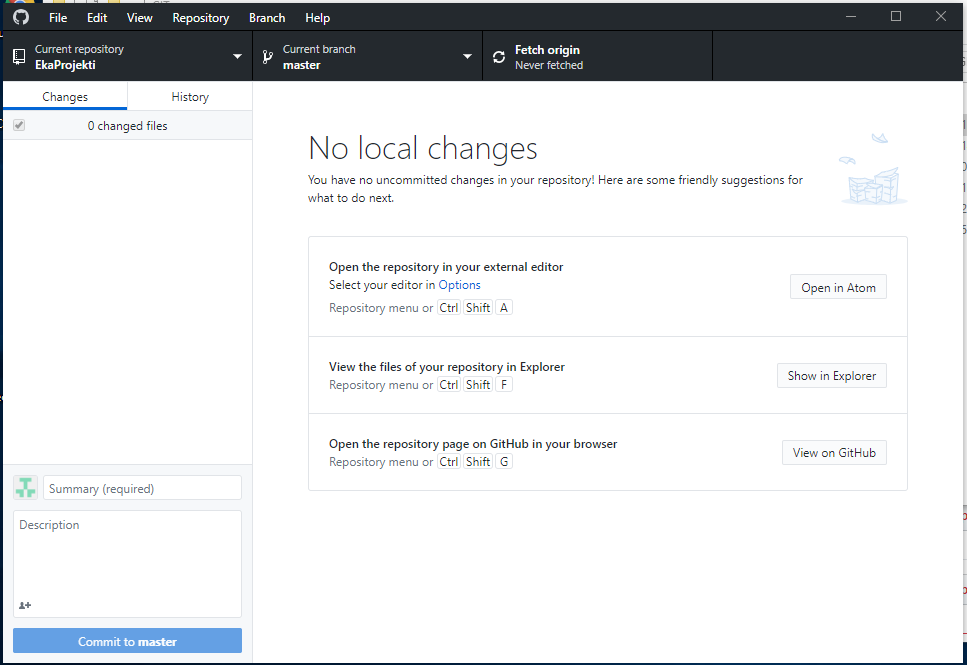
Open in Desktop-valinta avaa GitHubDesktop.exe-sovelluksen, jos se on asennettu. näkymän, johon lisätään paikallisen levyn polku. Esimerkissä tiedot viedään Windows-järjestelmän C:\xampp\htdocs\GIT-kansioon.



Local path-kenttään voidaan määrittää mikä tahansa kansio. Clone-painike tekee seuraavaksi kopion tiedoista ja kloonaa sen paikalliselle levylle. GitHub Desktop-käynnistyy ja avautuu näyttämään tiedot seuraavan kuvan mukaisesti.

## GitHub Desktop-käytön alkeet

Edellisen luvun kuvaamien vaiheiden GitHub Desktop-sovellus käynnistyy ja näyttää paikalliselle levylle kloonatun Repository-arkiston: EkaProjekti.



Ladattuja tietoja voit tarkastella kolmella tavalla:

1. Editorilla (nyt oletuksena on Atom-editori)
2. File Explorer-ohjelmalla
3. GitHubissa (alkuperäinen materiaali netissä)

Aloitetaan paikallisen Repository-arkiston README.md-tiedoston editointi Open in Atom-valinnalla.



Nyt README.md-tiedostoon on tehty muutos (lisärivi), joka voidaan päivittää GitHub-EkaProjekti-arkistoon. Muutos täytyy tietysti tallentaa ensin File-Save-komennolla paikalliseen kansioon. Tärkeää on ymmärtää, että tehty muutos menee paikallisen Repository-arkiston (EkaProjekti) master-haaraan! Editorista poistuttaessa (File-Exit-komennolla) avautuu GitHub Desktop-näkymä, jossa muutokset on dokumentoitu.

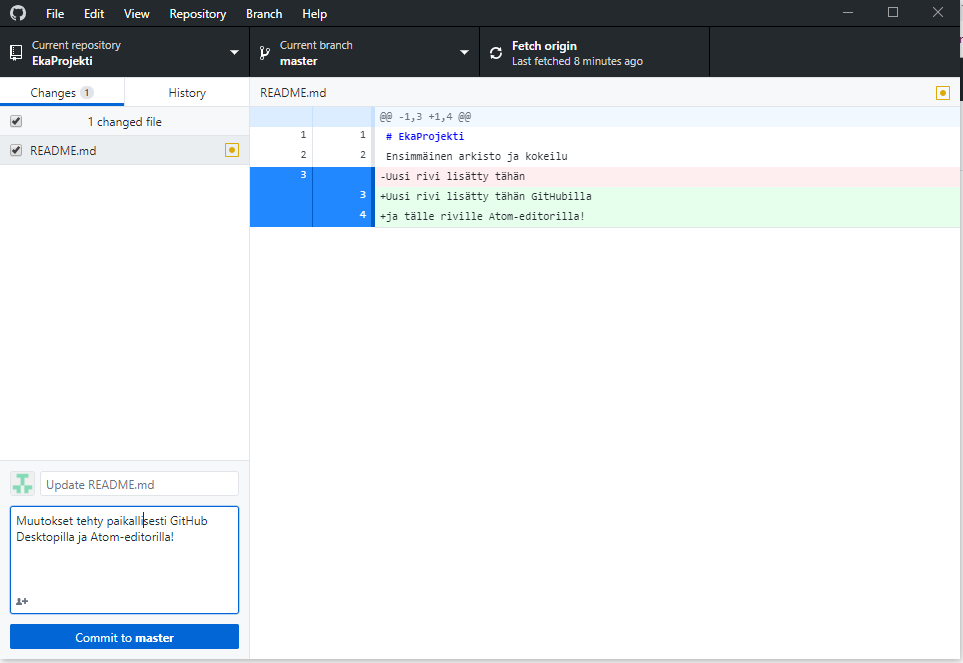


(tämä kuva on osatulostus näkymästä)

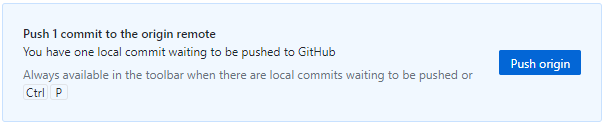
Paikalliset muutokset, GitHub Desktopissa tehdyt, päivittyvät GitHubin ottoburman/EkaProjekti-arkistoon painettaessa Commit to master-painiketta



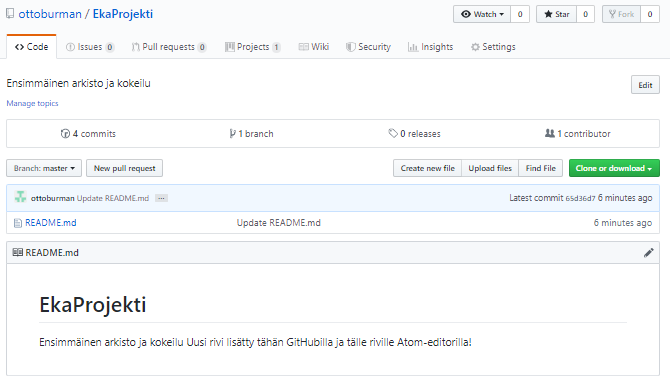
joka sijaitsee sivun vasemmassa alalaidassa. Tärkeää on myös muistaa kirjoittaa muistiinpanoja tehdyistä muutoksista alhaalla olevaan tietoikkunaan!



Kun Commit to master-painiketta on painettu, niin avautuu varmistusikkuna:



Tässä uudessa ikkunassa on painettava vielä **Push origin**-painiketta, joka vie (push) tiedot internetissä olevaan GitHubin Repository-arkiston master-haaraan. Tämä on esitetty seuraavassa kuvassa.



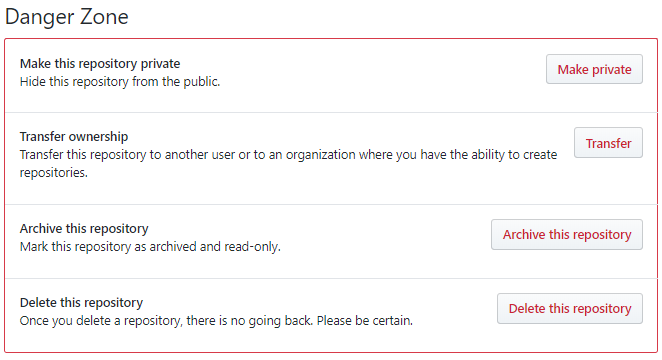
README.md-tiedoston tiedot ovat päivittyneet!

## Tarpeettoman GitHubin Repository-arkiston poistaminen

Tässä esityksessä on luotu Repository-arkisto jota ei enää tarvita ja se voidaan poistaa. GitHubissa tulee olla arkiston pääsivulla, kuten edellisessä kuvassa (ottoburman/EkaProjekti). Valitaan Settings-asetukset-lipuke oikeasta yläkulmasta:



Avautuvaa näkymää täytyy vierittää näkyvän sivun loppuun niin kutsutulle vaaralliselle alueelle (Danger Zone):



Poistaminen tapahtuu valitsemalla **Delete this repository-**Poista tämä arkisto-painike. Tämän jälkeen tulee erinäisiä varoituksia ja varmistuksia. *Lue varoitukset* – harkitse tarkkaan mitä olet tekemässä! Kyseistä Repository-arkistoa ei enää löydy GitHubin hakukomennoilla, mutta …



Huomaa, että tämä esimerkki on luonut kopion README.md-tiedostosta C:\xampp\htdocs\GIT-kansioon, jota käytetään paikallisella GitHub Desktop-sovelluksella. Palauttaminen ei ole kovin yksinkertaista, sillä järjestelmä tunnistaa poistetun arkiston sekä arkistojen haarojen commit-historian erot. Kiertotienä on, että arkistosta luodaan uusi versio ja uusi nimi, johon sitten haetaan (upload-komennolla) kansiosta päivitetty README.md-tiedosto. Tämä ei ole kovin tyylikäs ratkaisu – valitan. Ohessa vielä kuva todisteena.



Desktop-versio ei osaa siirtää tietoja internet-sovellukseen edellä mainituista seikoista johtuen.

# PHP-projekti

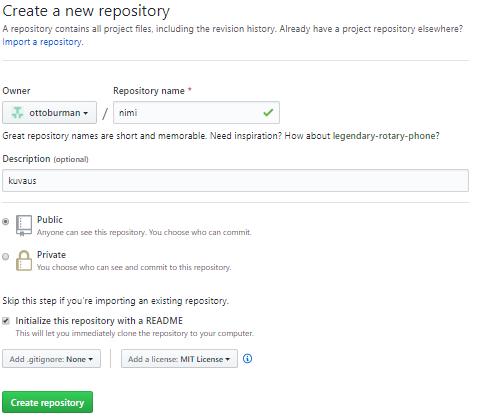
Tässä luvussa opitaan miten

* Luodaan ja käytetään Repository-arkistoa
* Aloitetaan ja hallinnoidaan Branch-haarautumia
* Tehdään muutoksia tiedostoon
* Työnnetään (Push) muutettu tiedosto GitHub-palveluun Commits-merkintöinä
* Avataan Pull Request-pyyntö
* Päivitetään Master Branch
* Kloonataan ja ladataan tiedostot omalle laitteelle
* Testataan tehdyt muutokset paikallisesti

Sovellusta kehitetään paikallisella XAMPP-Apache-palvelimella ja tuotantoversio on julkisella Internet palvelimella osoitteessa ottoburman.fi/oikealla/alalla.php.

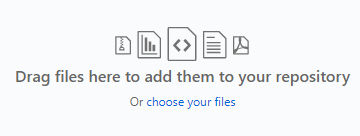
## Repo-arkiston luonti ja käyttö

Repository-arkisto on luotu edellisten lukujen ohjeiden mukaisesti seuraavin vaihein: 1) Kirjautuminen GitHubiin <https://github.com/login>, 2) siirrytään aloitustilanteeseen klikkaamalla koti/home-linkkiä: , 3) painetaan New-kuvaketta/näppäintä , avautuvaan näkymään kirjoitetaan Repo-arkistolle nimi, kirjoitetaan lisäkuvaus Description-kenttään, merkitään rasti kohtaan **Initialitse this…**, lisätään lisenssiehdot alasvetovalikosta **Add a licence:**, josta valitaan MIT License.



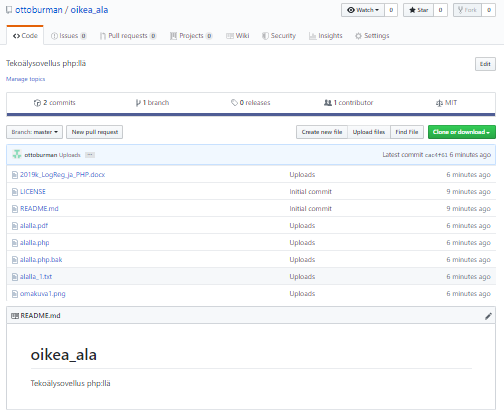
4) lopuksi painetaan vihreää **Create repository**-painiketta, jolloin uusi näkymä aukeaa.

5) uudessa näkymässä painetaan **Upload files**-painiketta , jolloin tiedostojen hakuikkuna avautuu. Tiedostot voidaan yksinkertaisesti vetää oman tietokoneen kansioista (File manager-ikkunasta) tähän avautuvaan pudotuslaatikkoon:



Tässä vaiheessa on hyvä opetella dokumentoimaan Commit-muutoksia tässä näkymässä oleviin kenttiin. Repo-luonnin lopputulos on esitetty seuraavalla sivulla.

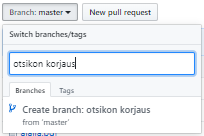
Syntynyt oikea\_ala-Repo-arkisto on GitHubissa, jossa tilanne näyttää seuraavalta (10.7.2019):



Arkisto on kaikille avoin, joten sinne voi mennä katsomaan nykytilannetta helposti kirjoittamalla selaimen komentoriville <https://github.com/ottoburman/oikea_ala> tai klikkaamalla linkkiä.

## Branch-haarautumien hallinnointia

Luodaan uusi haarautuma, jossa tavoitteena on korjata käyttöliittymän otsikko suuremmaksi. Painetaan kuvaketta **Branch: master**, jolloin avautuu ikkuna, johon merkitään uuden haarautuman nimi ja painetaan **Create branch: otsikon korjaus-**riviä kuten seuraavassa kuvassa on tehty:



Uuden haarautuman voi nähdä nyt **Branch: otsikon-korjaus-**painikkeesta,



joka myös samalla osoittaa, että nyt olemme tekemässä tähän työtilaan liittyviä muutoksia. Tuloksena on siis kopio master-päähaaran tiedoista.

Seuraavaksi muutetaan alalla.php-tiedoston koodia otsikon-korjaus-haarautumassa.

## Tiedoston editointi

Seuraavaksi korjataan PHP-tiedoston html-koodia niin että otsikon koko on aiempaa suurempi: html-osassa vaihdetaan <h3> suuremmaksi otsikoksi <h1> eli pääotsikoksi.

Repository-arkistoon pääsee myös selaimen komennolla suoraan kirjoittamalla komentoriville URL-osoitteen <https://github.com/ottoburman/oikea_ala/tree/otsikon-korjaus>.

Valitsemalla tiedostoluettelosta alalla.php-tiedoston klikkaamalla nimeä



Klikkaus avaa näkymän, josta päästään korjaamaan ohjelmakoodia, kun painetaan editointi-kynää tarkoittavaa kuvaketta:

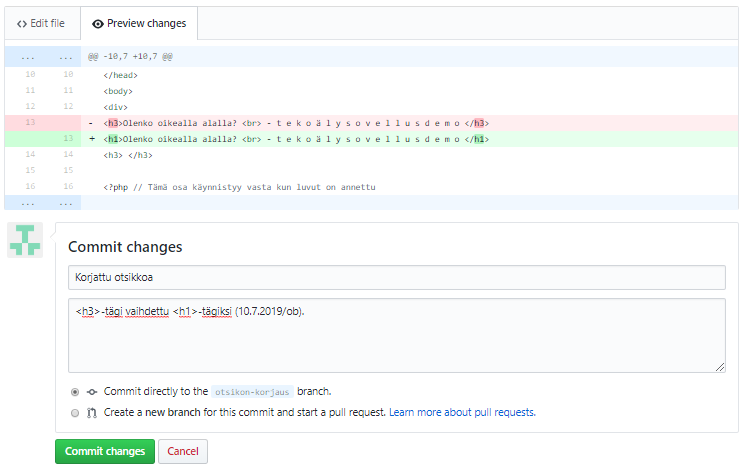


**Edit file**-lipuke näyttää tiedoston sisällön



## Commits-muutosmerkinnät

Riville 13 on nyt vaihdettu otsikon kokoa ilmaisevat merkinnät. Muutoksia voidaan tarkastella Preview changes-lipukkeen alta, jolloin ne näkyvät selvästi:

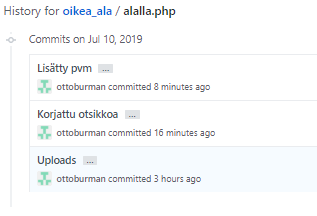


Commit-muutosmerkinnät ovat tässä helposti dokumentoitavissa ja esitetty ymmärrettävästi. Lopuksi muutokset ja dokumentointi kuitataan vihreällä **Commit changes**-painikkeella, jolloin palataan näyttämään tiedoston sisältö.

Nyt havaitaankin vielä korjaustarve: näyttäisi, että rivillä 14 on turhaa html-koodia. Toistetaan siis edelliset editoinnin vaiheet ja palataan takaisin näyttämään tiedoston sisältö:



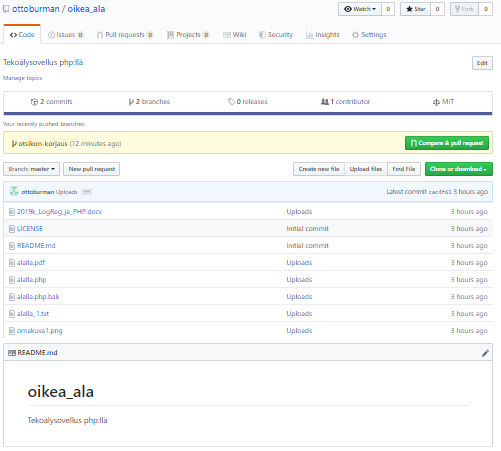
**History**-painike näyttää tehdyt muutokset:



Commits-korjaukset näkyvät selkeästi ja kuvasta käy selville, että tiedosto on tuotu (uploads), otsikkoa on korjattu ja päivämäärätieto on lisätty. Nyt voidaan palata takaisin oikea\_ala-Repository-arkistoon joko painamalla **<> Code**-lipuketta tai klikkaamalla ylhäällä olevaa arkistopolkua

.

Oikea\_ala-arkiston tiedot näyttävät nyt seuraavilta:

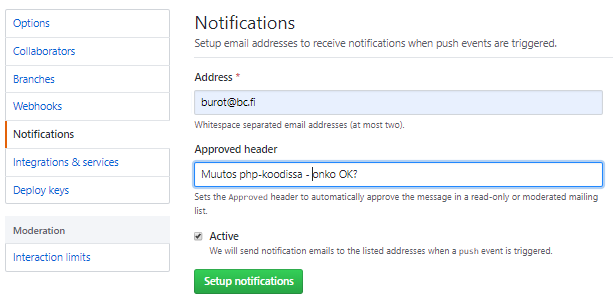


Tärkeää on huomata, että nyt olemme master-päähaarassa, joka näkyy **Branch: master**-painikkeen otsikosta. Seuraavaksi tehdyt muutokset hyväksytetään ja muutokset viedään master-päähaaraan.

## Pull Request-pyynnöt

### Notifikaatioluettelo

Aloitetaan tämä prosessi korjaamalla GitHubin Settings-asetuksia: Painetaan Settings-lipuke aktiiviseksi ja avautuvan sivun vasemmassa laidassa merkitään Notifications-valinta aktiiviksi klikkaamalla tekstiä. Nyt avautuu näkymä, johon voidaan merkitä vastaanottajia ilmoituksille.



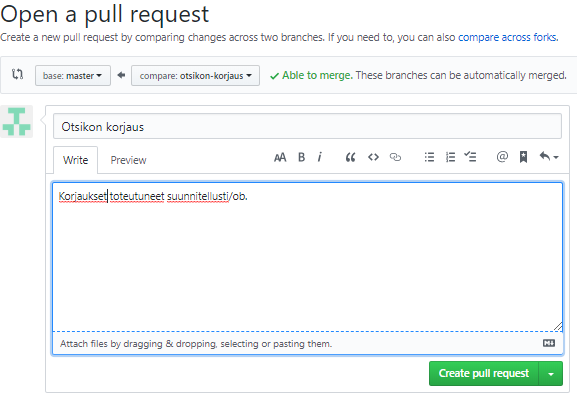
Muutokset astuvat voimaan, kun vihreää **Setup notifications**-painiketta on klikattu: Ilmoitukset tulevat nyt työpaikkani sähköpostiin! Paluu takaisin arkistoon tapahtuu **<> Code**-lipukkeen avulla.

### Open pull Request

Palataan takaisin edellisen sivun tilanteeseen tarkastelemaan Repository-arkistoa (**<> Code**-lipuke) ja avautuvassa näkymässä painetaan **Compare & pull request**-näppäintä:



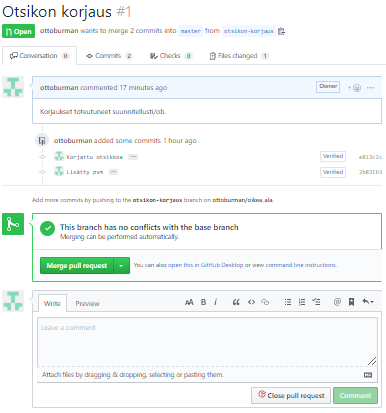
Seuraavaksi avautuu Open pull request-ikkuna:



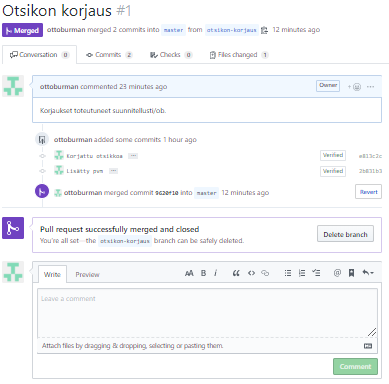
Tähän ikkunaan voi lisätä erilaista tietoa, mutta nyt siihen on kirjoitettu lyhyt tiedonanto. Vihreä **Create pull request**-näppäin käynnistä pyynnön.

## Master Branch päivitys

**Create pull request**-näppäin avaa tietoikkunan tehdyistä muutoksista:



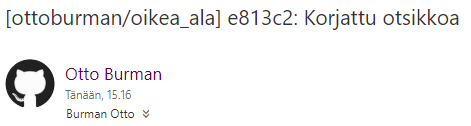
Merge pull Request-painike yhdistää haarautumaien tiedot ja seuraava näkymä näyttää tilanteen:



Keskellä on huomautus, jossa todetaan, että yhdistämien on onnistunut ja tämä haara voidaan poistaa. Tässä esimerkissä ei haarautumaa poisteta **Delete branch-**painikkeella (ainakaan tässä vaiheessa).

### Sähköpostin viestit eli GitHub-notifikaatiot

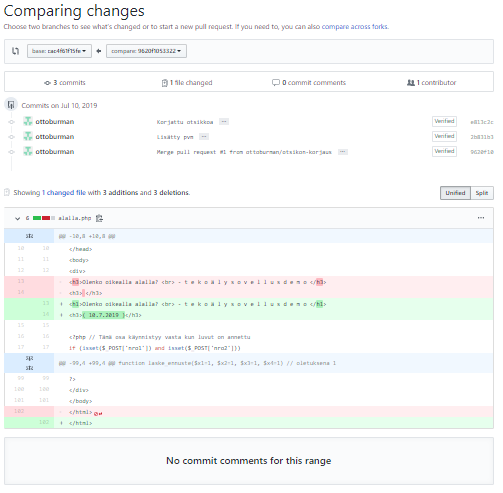
Lopuksi tarkistetaan vielä sähköpostin perille meno. Oheinen kuva osoittaa postin saapuneen.



Sähköposti pitää sisällään linkit joiden avulla voidaan tarkastella tapahtumaketjua ja tuloksia. Sähköpostin viimeiset rivit kertovat liittämisen onnistuneen

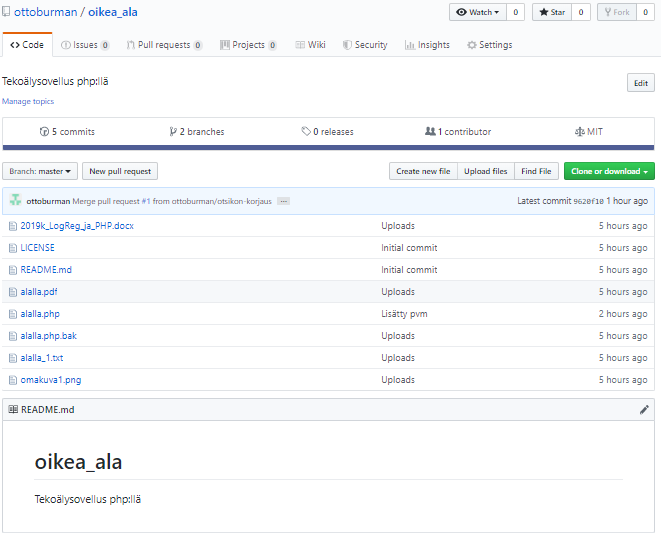
Merge pull request #1 from ottoburman/otsikon-korjaus  
Otsikon korjaus  
Compare: <https://github.com/ottoburman/oikea_ala/compare/cac4f61f15fe...9620f1053322>

Napsauttamalla linkkiä saadaan esille näkymä, josta selviää alalla.php-tiedostoon tehdyt muutokset



Näkymästä selviää kehitystyön nykytila. Seuraavaksi vielä viedään syntynyt uusi sovellus paikalliselle levylle ja testataan sen toimivuus.

## Muutokset tuotantoympäristöön

Tässä kuvassa on näkyvissä oikea\_ala-Repository-arkisto. 

Tietojen päivitys näkyy alallla.php-tiedoston kohdalla tekstinä (Lisätty pvm). Klikkaamalla tekstiä siirrytään näkymään, jossa nähdään viimeisin muutos rivillä 14.

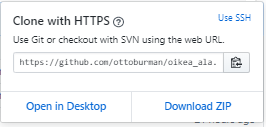


Näkymästä palataan takaisin **<> Code-lipukkeen** painalluksella.

Repository-arkiston näkymässä painetaan **Clone or download**-painiketta



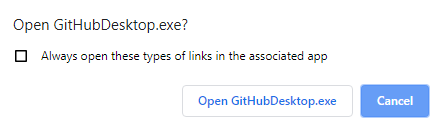
jolloin avautuu valintaikkuna:



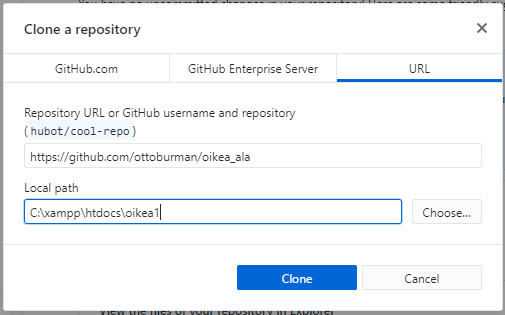
Open in Desktop-valinta avaa yhteyden paikalliseen tietokoneeseen ja sen avulla siirtyvät Repository-arkiston kaikki tiedot valittuun kansioon. Download ZIP-valinta lataa arkiston tiedon pakattuna z-tiedostona, joka voidaan tietysti purkaa minne tahansa osoitettuun hakemistoon. Näitä molempia tapoja käsitellään seuraavissa kappaleissa lyhyesti.

### Open in Desktop-kloonaus

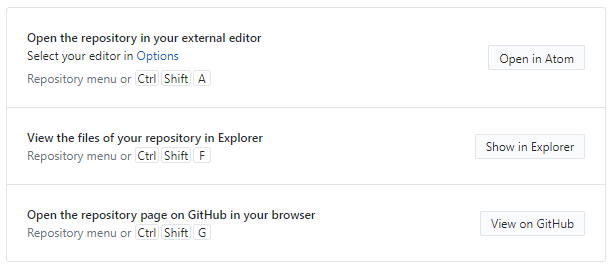
Open in Desktop-valinta pyytää vielä varmistamaan, että avaaminen tapahtuu GitHubDesktop.exe-sovelluksella. Mikäli sovellusta ei ole asennettu, GitHub ohjaa asentamaan sovelluksen oikeasta nettiosoitteesta. Seuraavassa ikkunassa valitaan **Open GitHubDesktop.exe-**painike.



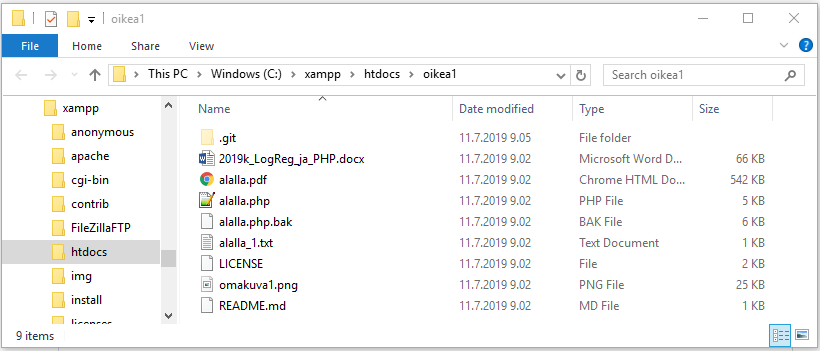
Seuraavaksi kerrotaan mihin kansioon tiedot kloonataan. Tässä esimerkissä tiedot laitetaan XAMPP-järjestelmän htdocs-kansion alikansioon, jolle on annettu nimeksi oikealla1. Kansiolle on annettu versionumero, jotta tiedot eivät korvaa mahdollisesti vanhoja versioita.



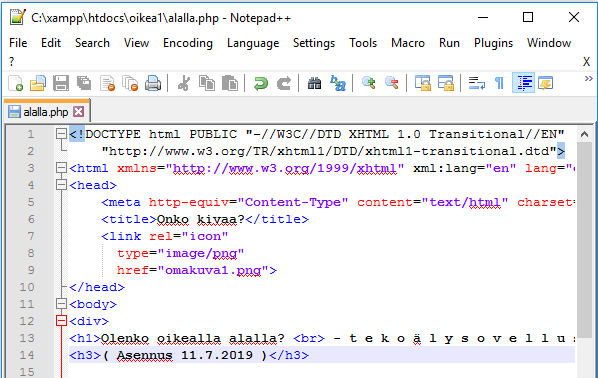
Tietojen siirryttyä GitHubDesktop-sovellus käynnistää ja näyttää millä tavoin tietoja voidaan alkaa käsitellä: Open in Atom (avataan Atom-editorilla), Show in Explorer (näytetään tiedostohallista-sovelluksella) tai View in GitHub (näytetään tiedot internetissä GitHub-sivustolla).



Tässä esimerkissä avataan tiedostohallinta / File Explorer-näkymä kansioon:

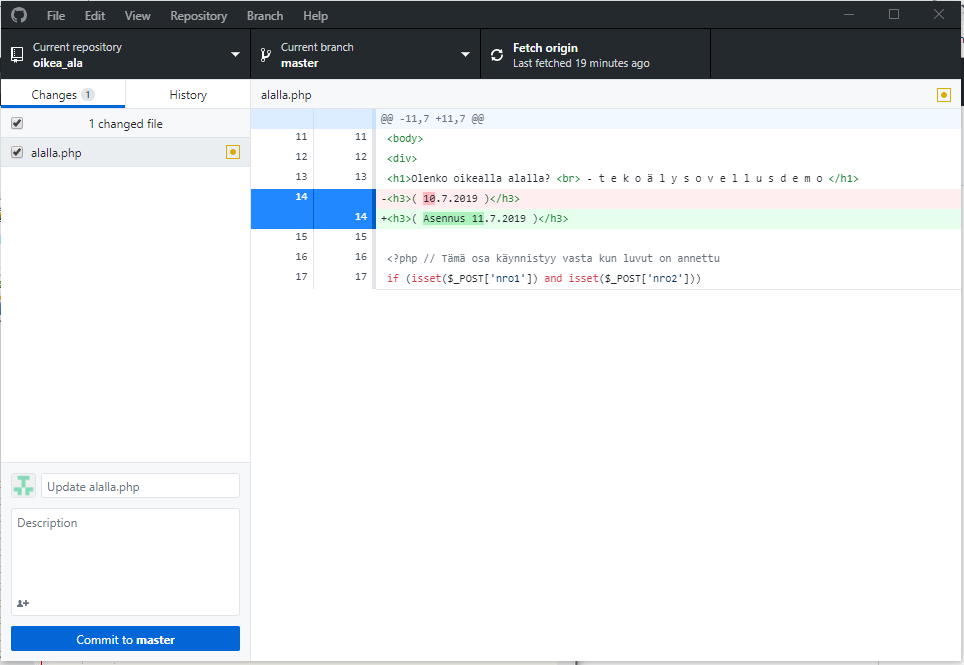


Nyt tällä kertaa editoidaan alalla.php tiedostoa Notepad++-editorilla:

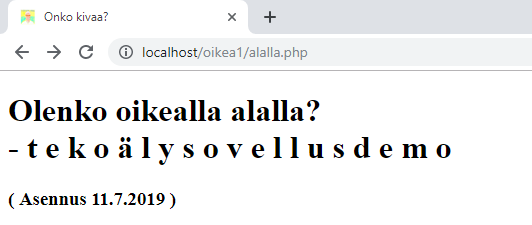


Kuvasta näkyy, että rivillä 14 on nyt uusi teksti ja päivämäärä. Seuraavaksi siirrytään automaattisesti aktivoituneen GitHubDesktop-sovelluksen näytölle klikkaamalla sitä.

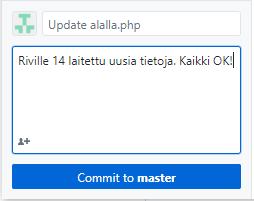
GitHubDesktop-sovellus on automaattisesti päivittänyt muutokset. Tämä on esitetty seuraavassa kuvassa.



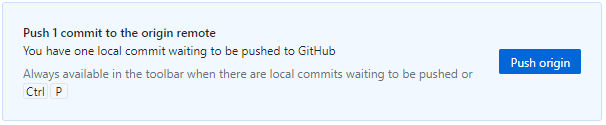
Seuraavaksi halutaan vielä varmistua muutoksien toimivuudesta: testataan muokattua sovellusta avaamalla tiedosto selaimessa localhost/oikea1/alalla.php-komennolla. (XAMPP-ympäristön Apache-palvelimen tulee olla tietysti käynnissä).

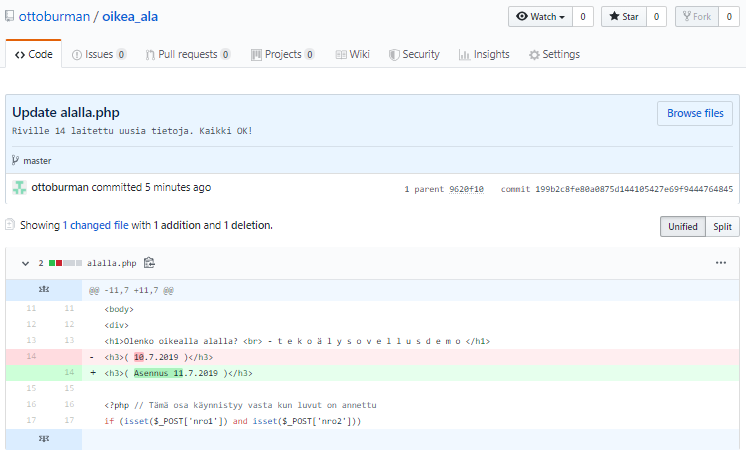


Päivitykset näkyvät selaimessa toimivan oikein, joten kirjataan tiedot GitHubDesktop-sovelluksen commit-muutosmääritykseen. Painetaan vielä Commit to Master-painiketta.



Seuraavaksi GitHubDesktop pyytää vahvistuksen päivityksen viemisestä (Push origin) GitHub-sivustolle:

 Painetaan Push origin-painiketta, jolloin tiedot päivittyvät ottoburman/oikea\_ala päähaaraan. Tämä voidaan käydä toteamassa GitHubissa, joka näyttää muutokset:

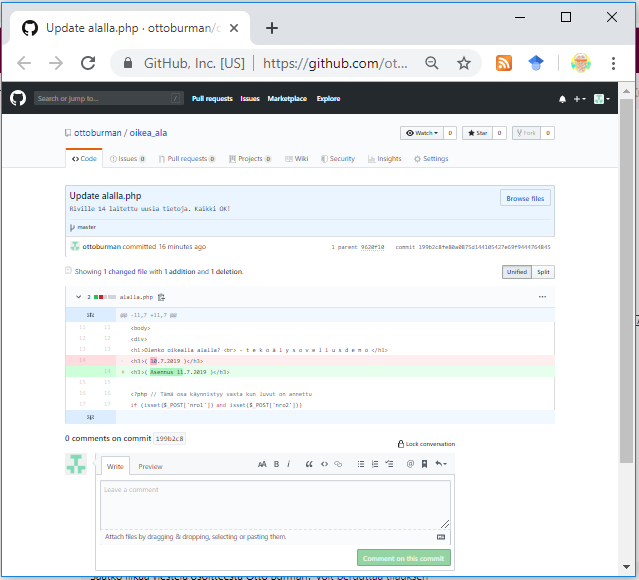


Lopuksi käydään vielä tarkistamassa sähköpostiin tulleet ilmoitukset.

Seuraavassa kuvassa on näkyvissä GitHubin lähettämä viesti. Sähköpostin kuvakaappauksesta näkyy esimerkiksi Repository-arkiston sijainti <https://github.com/ottoburman/oikea_ala>, johon päästään osoitetta klikkaamalla.



Mutta tärkein linkki on sähköpostissa annettu Commit-osoite, jota klikkaamalla avautuu sähköpostin vastaanottajalle yhteys muutostietoihin (tämä näytettiin jo aikaisemmin).



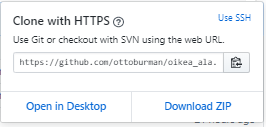
Sähköpostin vastaanottajalla on nyt antaa oma panoksensa kehitystyöhön ja tässä avautuvan sivun alaosassa tarjotaan mahdollisuutta kommentoida tehtyjä muutoksia!

### Download ZIP-kopiointi

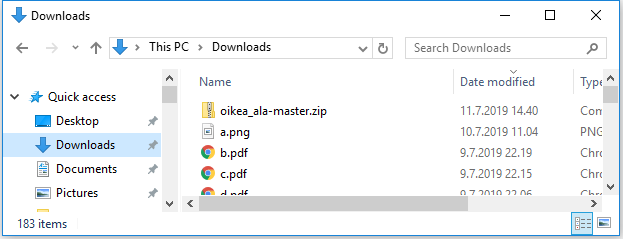
Repository-arkiston näkymässä painetaan **Clone or download**-painiketta



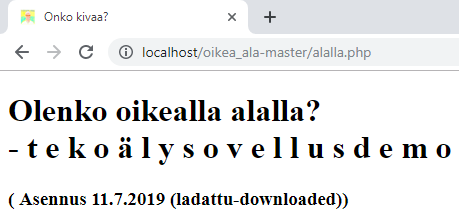
jolloin avautuu valintaikkuna, josta



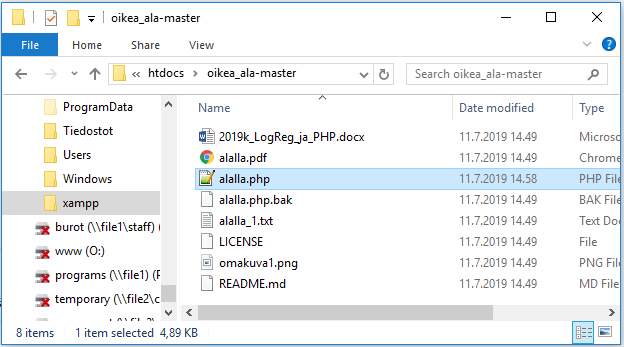
Download ZIP-valinta lataa arkiston tiedot zip-tiedostona suoraan Windows-ympäristössä automaattisesti määritettyyn ladatut tiedostot-kansioon (Downloads), josta se voidaan purkaa minne tahansa hakemistoon tai kansioon.



Seuraavaksi kansio on viety C:\windows\xampp\htdocs\alikansioksi. Alalla.php-tiedostoon on lisätty tekstiä riville 14 ”(ladattu-downloaded)”. Lopputulos saadaan selaimella näkyviin komentorivikomennolla <http://localhost/oikea_ala-master/alalla.php>, joka on esitetty seuraavassa kuvassa.



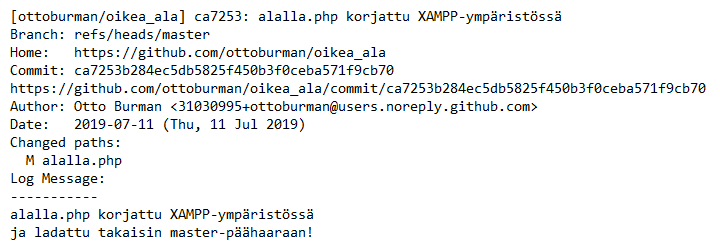
Lopuksi tarkastellaan vielä oikea\_ala-master-kansiota ja sen tiedostoja. Tarkastelu osoittaa, että kansiosta puuttuu git.-kansio, joka syntyi edellisessä tiedostojen siirron (GitHub-kloonauksen) yhteydessä.



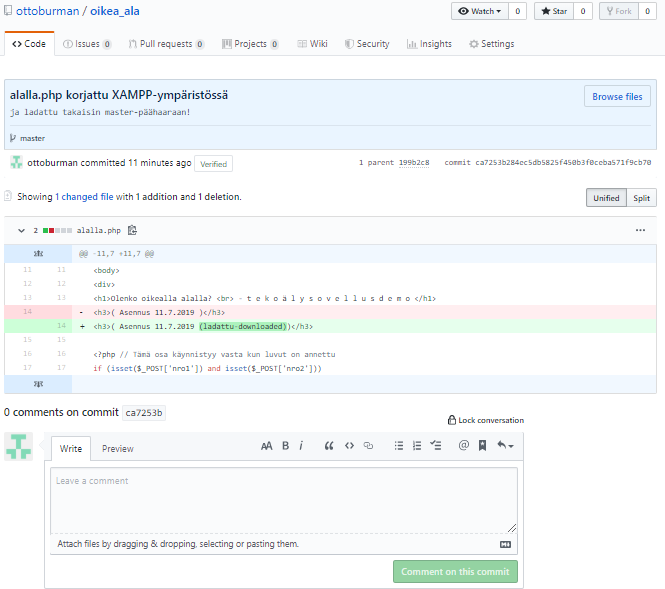
Lopuksi voidaan siis todeta, että tiedostojen lataaminen Download ZIP-valinnalla tulee tehdä esimerkiksi silloin kun haluaa käyttöönsä itselleen ohjelmakoodi talteen tai edelleen kehitettäväksi jossain muussa kehitysympäristössä.

Tietysti tämän sovelluksen päivittäminen GitHubin oikea\_ala-Repository-arkistoon on varsin yksinkertaista: GitHubissa painetaan Upload files-painiketta ja raahataan tiedosto arkistoon.

Tiedoston siirto takaisin GitHubiin dokumentoituu ja tiedostoa voi tarkastella Commit-muutosmerkinnän kautta. Sähköpostiin tulee myös tästä viesti, joka on esitetty seuraavassa kuvassa:



Klikkaamalla sähköpostissa Commit-linkkiä päästään jälleen kommentoimaan kehitystyötä ja sen etenemistä. Klikkaamalla avautuva näkymä on esitetty seuraavassa kuvassa:



Keskellä näkymää (Showing 1 changed file with 1 addition and 1 deletion) muutosraportissa näkyy tehty lisäys riville 14.

GitHub huolehtii kehitystyön dokumentoinnista ja vuoropuhelusta kaikkien kehittäjien kesken!

# Hankittu ja tavoiteltu osaaminen

Tekemällä kaikki lukujen 1 ja 2 vaiheet ohjeiden mukaisesti opitaan luomaan kehitysprojektille arkisto ja tekemään kommenttipyyntöjä GitHubissa! Kehitysprojekti osataan myös kopioida (kloonata) paikalliselle Windows-tietokoneelle ja laittaa sovelluksen komponentit haluttuun kansioon. Tehdyt muutokset osataan myös päivittää alkuperäisiin arkistoihin.

Harjoituksen avulla on nyt saavutettu varsin kattava osaaminen ja

huolellinen oppija osaa

* luoda itselleen GitHub-käyttäjätunnukset
* asentaa paikallisen GitHub Desktop-sovelluksen
* asentaa ja ottaa käyttöön esimerkiksi Atom-editorin
* ottaa GitHub-ympäristön käyttöönsä (github.com/login)
* luoda avoimen lähdekoodin Reository-arkiston GitHub-ympäristöön
* aloittaa ja hallinnoida uutta Branch-haarautumaa
* muuttaa tiedostoja ja toteuttaa commit-merkinnät
* avata ja yhdistää pull request-pyyntöjä
* tarkastella syntyneitä muutoksia GitHub-sovelluksessa
* kloonata projektin Repository-arkisto (download) omalle paikalliselle levylle
* tehdä paikalliseen Repository-arkistoon muutoksia
* siirtää paikalliset muutokset (push) takaisin (upload) GitHub-ympäristöön, jolloin GitHubin Repository-arkiston tiedot päivittyvät!
* poistaa tarpeeton GitHubin Repository-arkisto

# Tekijän yhteystiedot

|  |  |  |
| --- | --- | --- |
| C:\Users\burot\AppData\Local\Microsoft\Windows\INetCache\Content.Word\vk_Burman_kuva2019_440x580.jpg | KL, ekonomi Otto Burmanin kasvatustieteen alaan kuuluva väitöskirja *Oppimisilmapiirin tekijöinä ilo, psykologinen omistajuus sekä yrittäjämäinen oppiminen – oppimisen ilon rakenneyhtälömallinnus*ontarkastettu 19.6.2019 Itä-Suomen yliopiston filosofisessa tiedekunnassa. Vastaväittäjänä tilaisuudessa toimivat dosentti Jaana Seikkula-Leino (Turun yliopisto) ja kustoksena professori Liisa Karlsson (Itä-Suomen yliopisto). |  |

Oppimisilmapiirin tekijät

Oppimisen ilon ilmapiirin edistämisen ja opetuksen kehittämisen tavoitteet voidaan johtaa ja muotoilla kehittämisen poluksi, jossa ilmapiirin parantamista lähestytään opettajan työhön liittyvistä tavallisista tehtävistä kuten ohjaaminen, opettaminen ja mahdollistaminen.

KL **Otto Burmanin** väitöskirjatutkimuksessa selvisi, että *oppimisen iloa* koetaan, kun oppilas saa tukea opintoihin ja oppimiseen, kun hän itse viihtyy ja muut viihtyvät, kun hän omasta mielestään kehittyy ja kun hänen ongelmaratkaisutaitonsa paranevat. *Oppimisen psykologista omistajuutta* koetaan kun (oppimis)tehtävät ovat omakohtaisia ja itselle sopivia, kun kuuluu ryhmään, kun suunnittelee omaa työtään, kun toteuttaa tehtäviä itseään varten, kun luottaa itseensä ja tuntee oppivansa mitä vain. *Yrittäjämäistä oppimista* koetaan, kun oppilas viihtyy uusien asioiden parissa, kun hän haluaa nähdä, kokea ja kehittää jotain uutta, kun hän panostaa haastaviin tehtäviin, kun hän tunnistaa osaamisensa ja kun hän on halukas oppimaan uutta. Näihin kolmeen ilmiöön vaikuttaa ylemmän tason tekijä, jota voidaan kutsua oppimisilmapiiriksi.

Oppijoiden tuntemukset on nyt saatu paremmin näkyviksi ja samalla saatu konkreettisiksi kehittämisen kohteiksi. Tuloksena ovat muun muassa syntyneet ketteriin kehitysmenetelmiin perustuvat oppiScrum-opetusmenetelmät (oppiscrum.fi).

Tutkimuksen pohdintaosassa peilataan tulosten ajankohtaisuutta tarkastelemalla lyhyesti viime vuosina tapahtuneita valtion koulutusmäärärahojen leikkauksia, ammatillisen koulutuksen läpäisyn ja keskeyttämisen kehitystä, koulupudokkaiden syrjäytymistä työelämästä ja nuorten jaksamisongelmia.

**Tuloksista ovat raportoineet esimerkiksi YLE ja sanomalehdet:**

<https://areena.yle.fi/1-50203440?seek=85>

<https://yle.fi/uutiset/3-10840138>

<https://digilehti.karjalainen.fi/titles/karjalainen/3544/archive> (Uutiset Karjalainen etusivu

ja A8 19.6.2019)

**Tutkimus on luettavissa ja haettavissa**

<http://ottoburman.fi/lectio.pdf> (Akateeminen esittely)

<http://ottoburman.fi/documents/Burman_yhteenveto.pdf> (YHTEENVETO)

<http://epublications.uef.fi/julkaisutyypit/vaitoskirjat/>

<http://epublications.uef.fi/pub/urn_isbn_978-952-61-3106-1/urn_isbn_978-952-61-3106-1.pdf>

**Lisätietoja annan mielelläni:**

Otto Burman, p. 044 77 56 437, ottoburman(at)gmail.com, ottoburman.fi