



Ohjelmistorobotiikka

Raine Kauppinen

Tämä materiaali on julkaistu lisenssillä

[CC BY-NC-SA 4.0](#)

SeAMK 

SEINÄJOEN AMMATTIKORKEAKOULU
SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES



Tausta

Tämä materiaali on tuotettu osana Seinäjoen ammattikorkeakoulun Ohjelmistosuunnittelu-hanketta, jolle rahoituksen on myöntänyt Jatkuvan oppimisen ja työllisyyden palvelukeskus.

Palvelukeskuksen tehtävänä on edistää työikäisten osaamisen kehittämistä ja osaavan työvoiman saatavuutta sekä vastata nopealla toiminnalla työmarkkinoiden äkillisiin rakennemuutoksiin. Palvelukeskuksen toimintaa ohjaavat opetus- ja kulttuuriministeriö sekä työ- ja elinkeinoministeriö.





Johdanto

Ohjelmistorobotiikka

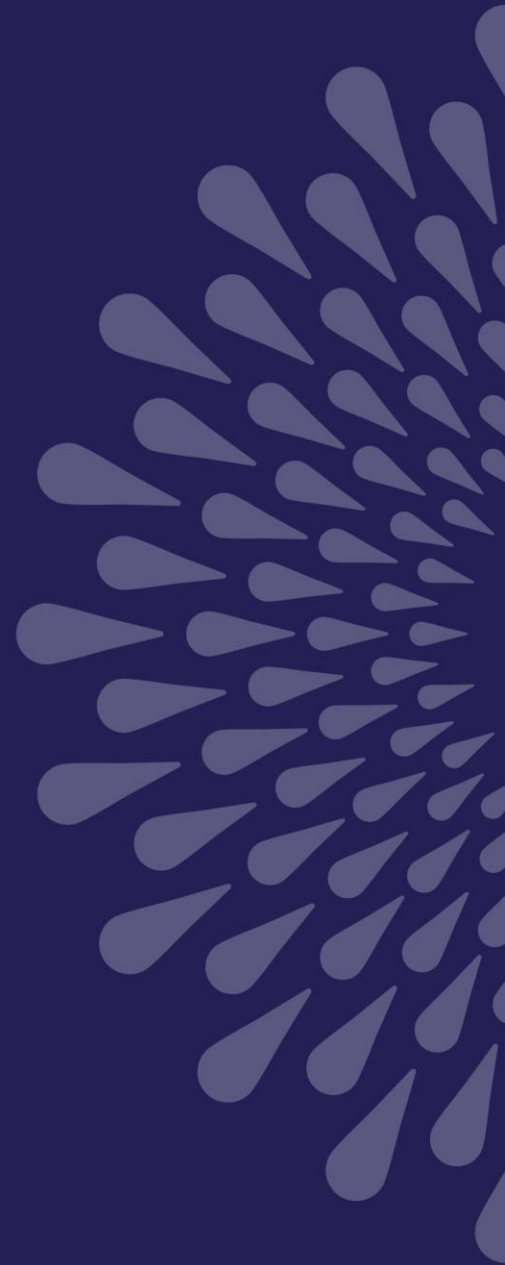
Raine Kauppinen

SeAMK 

SEINÄJOEN AMMATTIKORKEAKOULU
SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES



Ohjelmistorobotiikka ja RPA (robotic process automation)



Ohjelmistorobotiikka

- Ohjelmistorobotiikka on *tietokoneavusteista automaatiota*, jossa suoritetaan tehtäviä ohjelmistojen ja algoritmien avulla ilman ihmisen jatkuvaa osallistumista.
- Ohjelmistorobotiikassa yhdistyvät tietojenkäsittely, tekoäly ja robotiikka ja sen keskiössä on prosessien automatisointi ja tehtävien suorittaminen digitaalisessa ympäristössä.
- Usein ohjelmistorobotiikalla tarkoitetaan RPA:ta (Robotic Process Automation)



RPA (Robotic Process Automation)

- Ohjelmistorobotiikalla viitataan usein RPA:han (Robotic Process Automation), jolla tarkoitetaan ihmisen toimintaa ja käyttäytymistä eri sovelluksissa ja järjestelmissä simuloivia ohjelmistoja.
- RPA:lla saavutetaan usein nopeita hyötyjä, koska niillä voidaan automatisoida esimerkiksi olemassa olevia rutiininomaisia tehtäviä tai prosesseja kuten tietojen siirtoa ohjelmistosta tai järjestelmästä toiseen käyttämällä ohjelmistorobottia, joka käyttää ohjelmistoja tai järjestelmiä vastaavasti kuin ihminen (niiden ihmiselle tarkoitetun käyttöliittymän kautta).

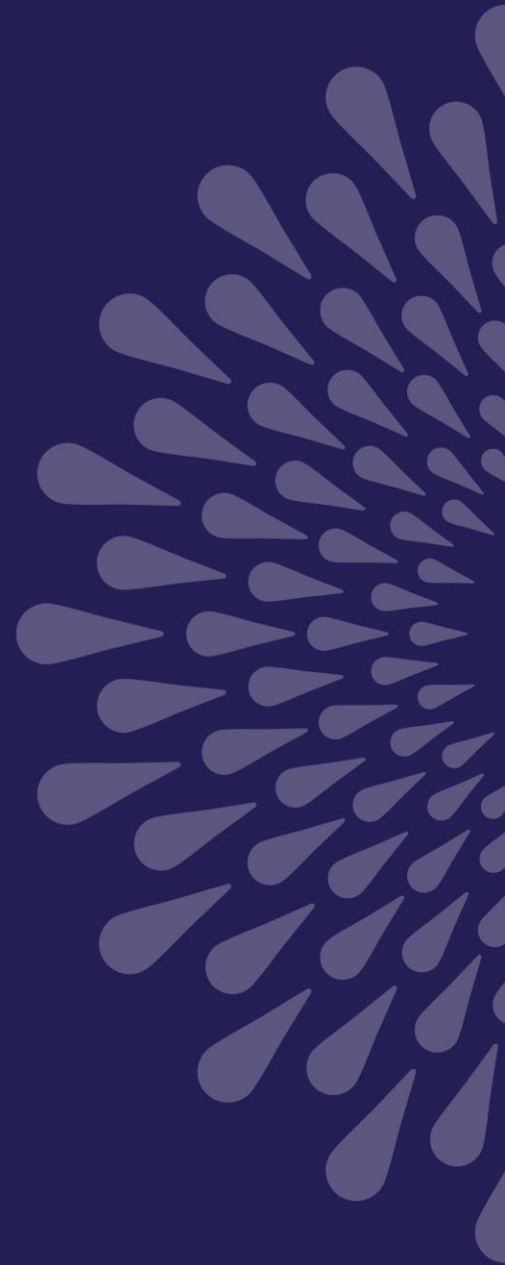


Ohjelmistorobotiikka ja RPA

- RPA on ohjelmistorobotiikan eräs osa-alue.
- Ohjelmistorobotiikassa voidaan kuitenkin myös suunnitella (alusta alkaen) prosessit ja tehtävät niin, että ne hoidetaan ohjelmistorobotiikalla joko kokonaan esimerkiksi algoritmien ja tekoälyn tai siten, että ihminen osallistuu prosessiin vain tiettyjen tehtävien osalta (esimerkiksi tekee päätöksen tai valinnan tai tarkistaa tuloksen).



Ohjelmistorobotiikan merkitys ja hyödyntäminen



Merkitys

- Koska ohjelmistorobotiikka on automatisointia, sillä voidaan tehostaa toimintaa ja siten saavuttaa esimerkiksi seuraavia hyötyjä:
 - Tehtävien suoritus nopeutuu ja prosessien läpimenoaika lyhenee.
 - Virheiden määrä (erityisesti inhimillisten virheiden määrä rutiininomaisissa tehtävissä) laskee.
 - Ihmisen aikaa vapautuu muihin tehtäviin (niihin, joita ei voi tai joiden automatisointi on vaikeaa).



Hyödyntämiskohteita

- Ohjelmistorobotiikkaa voidaan hyödyntää esimerkiksi seuraavissa tehtävien tai prosessien osa-alueissa:
 - Tiedon kerääminen (useasta lähteestä) ja käsittely (esimerkiksi muunnokset).
 - Tietokantojen päivittäminen.
 - Raporttien luonti.
 - Kommunikaatio (integraatio) eri järjestelmien välillä.



Hyödyntämisen tasoja

- Ohjelmistorobotiikkaa voidaan hyödyntää eri tasoilla (eri laajuudessa):
 - Organisaatioiden välillä esimerkiksi tietojen siirrossa ja synkronoinnissa.
 - Yksittäisessä organisaatiossa esimerkiksi yhteisten (liiketoiminta)prosessien tai –tehtävien automatisoinnissa.
 - Yksilötasolla esimerkiksi oman toiminnan (omien tehtävien tai niiden osien) automatisoinnissa.



Esimerkki: Organisaatioiden välillä

- Pohjois-Savon hyvinvointialue: Sosiaalihuollon vanhojen asiakastietojen siirtäminen ja arkistointi Kelan keskitettyyn sosiaalihuollon Kanta-palveluun sekä hallinnollisten tietojen arkistointi Pohjois-Savon omaan arkistoon.
- Lähde: Digital Workforce Services Oyj. Lehdistötiedote 10.8.2023. *Ohjelmistorobotiikkaratkaisu arkistoi sosiaalihuollon asiakastietoja Pohjois-Savossa.* Saatavissa esimerkiksi [STT Infossa](#).



Esimerkki: Yksittäisessä organisaatioissa

- Seinäjoen ammattikorkeakoulu on ottamassa käyttöön ohjelmistorobotin, joka tarkistaa, että opintojaksoista ja toteutuksista on ennen ilmoittautumisen alkamista Peppi-järjestelmässä vaaditut tiedot.
- Jos tietoja puuttuu, muistuttaa robotti ao. opettajaa sähköpostitse täydentämään puuttuvat tiedot.



Esimerkki: Yksilötasolla

- Seinäjoen ammattikorkeakoulussa henkilökunnan on kirjattava työtunnit Reportronic-järjestelmään kerran kuukaudessa.
- Yksittäinen opettaja voisi tehdä itselleen ohjelmistorobotin, joka kerää työtuntitiedot esimerkiksi sähköisestä kalenterista, Excelistä tai vastaavista lähteistä jos tarvittavat tiedot ovat saatavissa.

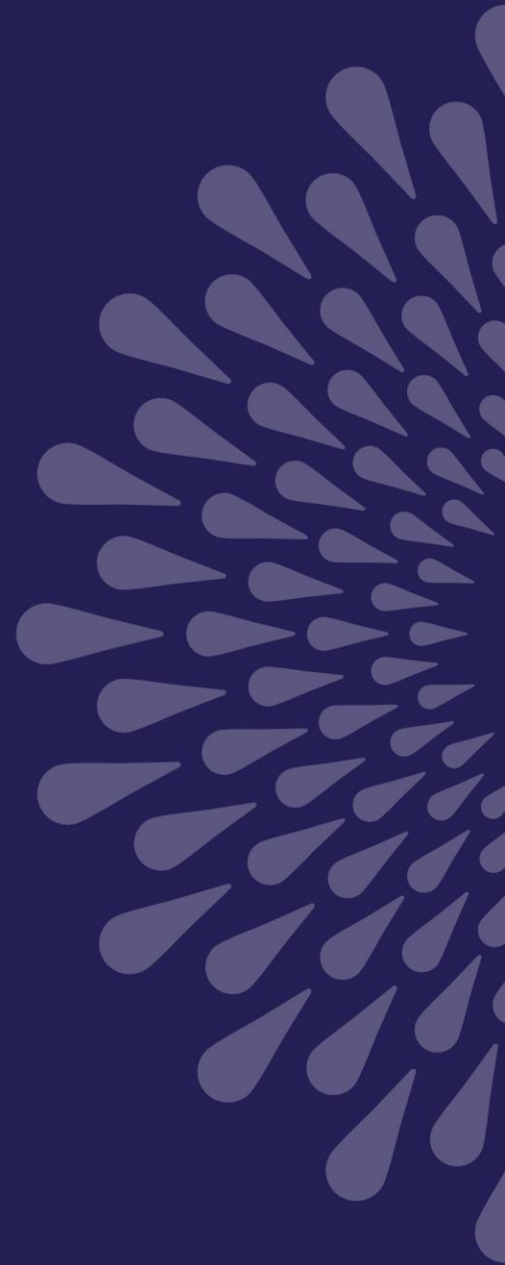


Harjoitus 1a: Orientaatio

- Tee tässä harjoituksen 1 a-kohta.
- Harjoitukset ovat erillisessä tiedostossa.



Ohjelmistorobotiikan käyttö- kehittämisympäristöt



Yksinkertaiset tai yleiset ympäristöt

- Ohjelmistorobotiikan esiasteena tai yksinkertaisen (RPA-)ohjelmistorobotiikan ympäristöinä voi pitää erilaisia makroja, nauhoittimia sekä ohjelmistokielen kirjastoja tai kehikkoja, joilla voidaan ohjelmallisesti ohjata syöttölaitteita ("kirjoittaa" syötteitä, "liikuttaa" tai "klikata" kohtaa ruudulla tms.).
- Lisäksi yleisohjelmointikielillä ja ympäristöillä on mahdollista toteuttaa ohjelmistorobottoja tai niiden kaltaisia ohjelmistoja ilman erityistä tukea ohjelmistorobottien tekemiseen.



Käyttöympäristöt

- Ohjelmistorobotit ovat suoritettavaa ohjelmakoodia, joilla suoritetaan automatisoituja prosesseja ja tehtäviä.
- Käytetystä ratkaisusta riippuen näille on käyttöympäristö, useimmiten hallintakäyttöliittymä ("valvomo" tms.), jossa robotteja (prosesseja) voi mm. käynnistää, pysäyttää ja seurata. Käyttöympäristö on usein, varsinkin laajemmissa tarkaisuissa, pilvipalvelu tai, esimerkiksi pienemmissä ratkaisuisissa tai yksilökäytössä paikallisesti asennettu.
- Lisäksi, jos ihminen osallistuu prosessiin tai tehtävään, käytetään usein käyttäjän laitteelle asennettavaa apuohjelmaa tai ohjelmistoa ("assistant" tms.).



Kehitysympäristöt

- Eri toimittajat tarjoavat eri teknologioihin (ohjelmointikieliin, -ympärisöihin, kirjastoihin ja kehikoihin) perustuvia ratkaisuja, joiden avulla ohjelmistorobotteja kehitetään ja testataan.
- Kehitysympäristöt voi jakaa matalan kynnyksen ratkaisuihin ("low-code" tai "no-code") ja ohjelmointiosaamista edellyttäviin ratkaisuihin.



Matalan kynnyksen ympäristöt

- Matalan kynnyksen ympäristöt eivät edellytä tai edellyttävät vähän ohjelmointiosaamista ja tarjoavat usein esimerkiksi visuaalisen ja esimerkki- tai käyttötapauspohjaisen rakentimen.
- Näitä käytetään usein esimerkiksi RPA-tyyppiseen ohjelmistorobottiikkaan.
- Eräs suosittu matalan kynnyksen ympäristö on [UiPath](#), joka tarjoaa mm. visuaalisen rakentimen ja käyttää ohjelmointikielenä Visual Basicia.



Ohjelmointiosaamista edellyttävät ympäristöt

- Ohjelmointiosaamista edellyttävät ympäristöt olettavat, että robottien kehittäjä tuntee tietyn tai tietyt ohjelmointikielet ja –teknologiat ja tarjoaa niihin esimerkiksi robottien ohjelmointiin tarkoitettun ympäristön, kehikon tai kirjastoja.
- Näitä käytetään usein ohjelmistorobottilähtöisten prosessien ja tehtävien toteuttamiseen.
- Eräs suosittu ohjelmointiosaamista edellyttävä ympäristö on [Robocorp](#), joka tarjoaa mm. Visual Studio Code -laajennoksen sekä Python ja RobotFramework API:t.



Harjoitus 1b: Orientaatio

- Tee tässä harjoituksen 1 b-kohta.
- Harjoitukset ovat erillisessä tiedostossa.





Käsitteitä

Ohjelmistorobotiikka

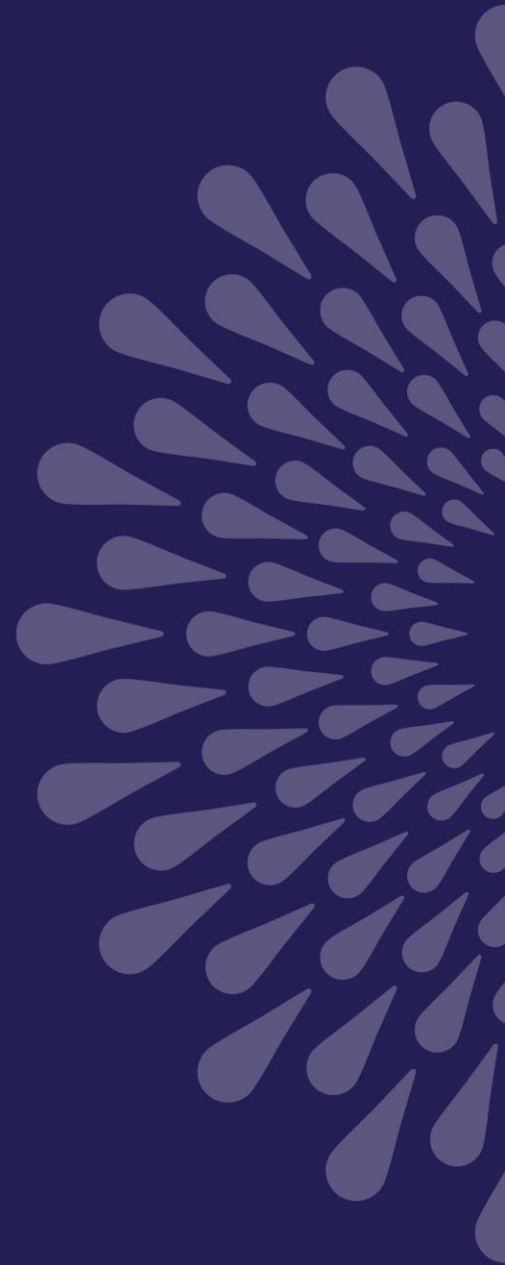
Raine Kauppinen

SeAMK 

SEINÄJOEN AMMATTIKORKEAKOULU
SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES



Ohjelmistorobotiikalla automatisoitava prosessi ja tehtävä



Prosessi ohjelmistorobotiikassa

- *Prosessi (process)* on kokonaisuus, joka automatisoidaan kokonaan tai osittain toteuttamalla ohjelmistorobotti tai useampi. Prosessista käytetään joskus myös termiä *työnkulku (workflow)*.
- Prosessi koostuu yhdestä tai useammasta tehtävästä ja sitä hallitaan ("orkestroidaan") käyttöympäristössä.

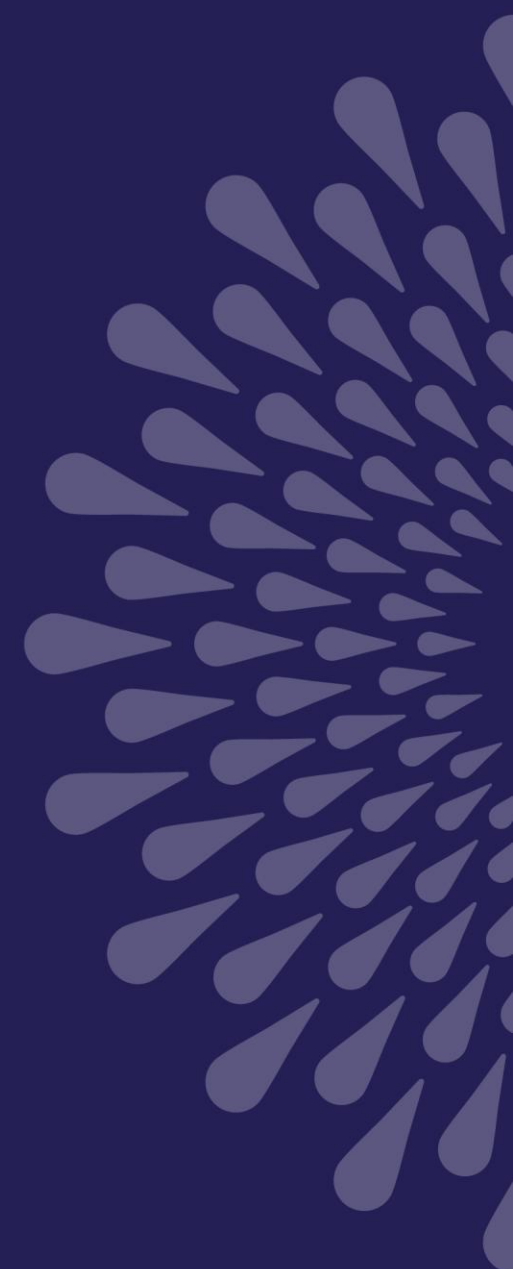


Tehtävä ohjelmistorobotiikassa

- *Tehtävä (task)* on toiminto tai sarja toimintoja, joita ohjelmistorobotti suorittaa yksi kerrallaan.
- Tehtävät voidaan käynnistää erikseen, joten tehtävä on ohjelmistorobotiikassa pienin toteuttava yksikkö, ts. jokaisen ohjelmistorobotin toteutus sisältää vähintään yhden tehtävän.



Teköällyn hyödyntäminen - agentti ja toimenpide



Agentti ohjelmistorobotiikassa

- *Agentti (agent)* on kielimallia (tekoälyä) hyödyntävä kokonaisuus, jonka avulla ohjelmistorobotti suorittaa (act) yhden tai useamman toimenpiteen.
- Agenttina toimiva ohjelmistorobotti pystyy kielimallin avulla päättellemään (reason) ja toimimaan yhteistyössä (collaborate) luonnollisella kielellä.

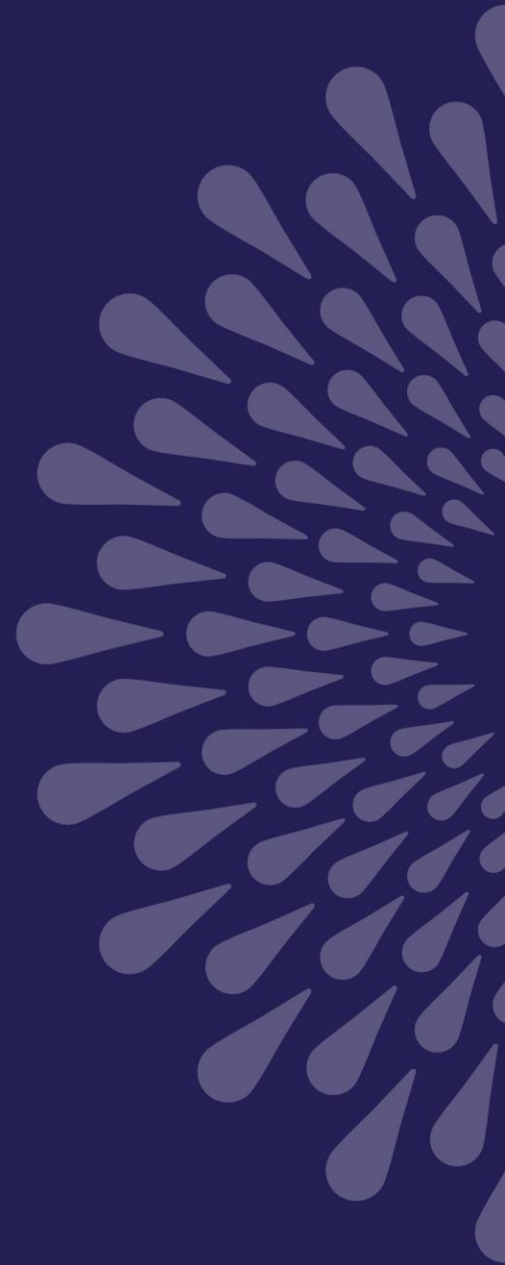


Toimenpide ohjelmistorobotiikassa

- *Toimenpide (action)* on toiminto tai sarja toimintoja, joita ohjelmistorobotti suorittaa käyttäen kielimallia (tekoälyä).
- Toimenpide voi kielimallin käytön lisäksi suorittaa tehtäviä. Kielimallin hyödyntäminen edellyttää usein palvelimen käyttöönottoa.



Ohjelmistorobotti ja sen käyttö sekä kehittäminen



Ohjelmistorobotti

- *Ohjelmistorobotti (software robot)* on kokoelma valitulla ohjelmointikielellä tai –tekniikoilla toteutettua ohjelmakoodia.
- Ohjelmistorobotin ohjelmakoodi kuvaa toimenpiteet, jotka tekemällä kukin tehtävä suoritetaan.
- Ohjelmakoodissa käytetään useimmiten runsaasti kehittämisympäristön tarjoamia kirjastoja eli valmista ohjelmakoodia, joka on tarkoitettu ohjelmistorobottien rakentamiseen.



Valvottu ja valvomaton ohjelmistorobotti

- *Valvottu (attended)* ohjelmistorobotti on vähintään jossakin kohdin eli jonkin tehtävän osalta käyttäjän ohjaama (se voi esimerkiksi pyytää tietyissä kohdissa varmistusta käyttäjältä, se voi olla manuaalisesti käynnistettävä tms.).
- *Valvomaton (unattended)* on ohjelmistorobotti itsenäinen, eikä tarvitse käyttäjän ohjausta (se voi olla esimerkiksi ajastettu ja suorittaa jonkin prosessin tietyin väliajoin).



Käyttö- ja kehittämisympäristö

- *Käyttöympäristö (tuotanto, production)* on ympäristö, jossa valmiita ohjelmistorobotteja hallinnoidaan (mm. ajastetaan, suoritetaan ja valvotaan). Käyttöympäristö voi olla pilvipalvelu, organisaation omassa ylläpidossa ja palvelimilla tai yksittäisellä tietokoneella tilanteesta riippuen.
- *Kehittämisympäristö (kehitys, development)* on ohjelmointiympäristö, jossa ohjelmistorobotteja rakennetaan ja valitulla välineillä, ohjelmointikielellä ja kirjastoilla. Ohjelmistorobotteja (tehtäviä) voi myös suorittaa (testata) kehittämisympäristössä.



Kirjastot ohjelmistorobottien kehittämiseen

- *Kirjastot (libraries)* ovat kehittämisympäristökohtaisia (esimerkiksi ohjelmointikielikohtaisia) valmiita ohjelmakoodoja ohjelmistorobotin tarvitsemien tyypillisten toimintojen toteuttamiseen.
- Kehittämisympäristö tarjoaa useimmiten peruskirjastot esimerkiksi tehtävien määrittelyyn ja lokitietojen tuottamiseen. Lisäksi keskeisiä ovat ajuri(kirjasto)t.



Ajurit ohjelmistorobotiikassa

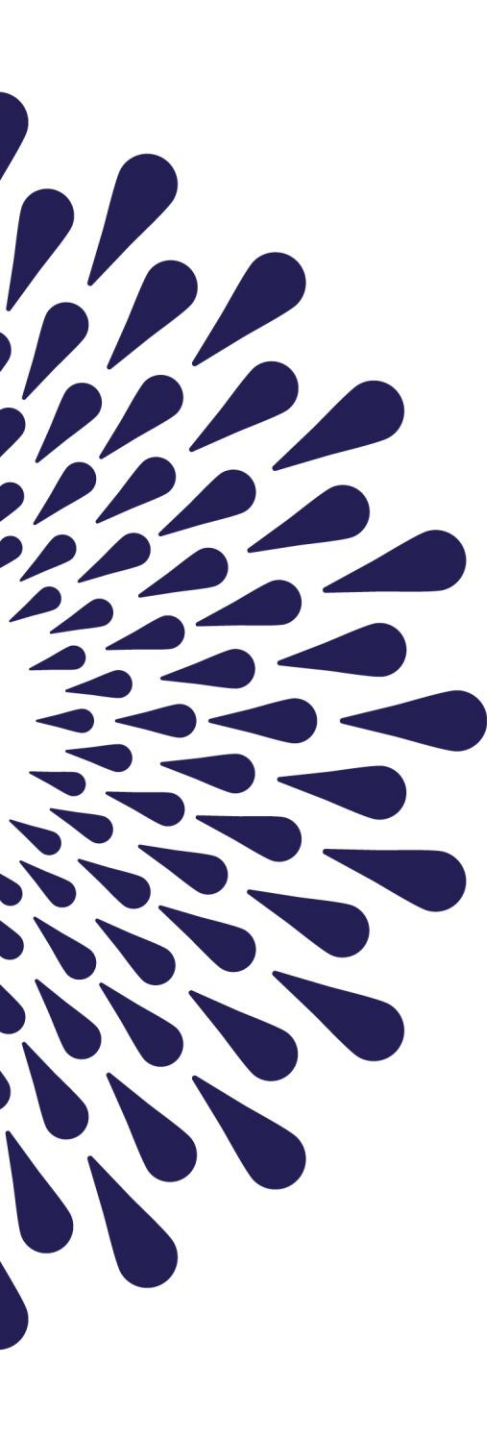
- *Ajurit (drivers)* ovat kirjastoja, joiden avulla ohjelmistorobotti käyttää käyttöjärjestelmää (esimerkiksi avaa jonkin ohjelman Windowsissa) tai ohjelmaa (esimerkiksi avaa jonkin www-sivun johonkin selaimeen).
- Ajurit mahdollistavat käyttäjän toiminnan simuloimisen ohjelmistorobotilla, eli ne ovat RPA-tyyppisen ohjelmistorobotiikan perusta. Niihin liittyvät keskeisesti paikantimet.



Paikantimet ohjelmistorobotiikassa

- *Paikantimet (locators, joskus myös valitsimet, selectors)* ovat ajurien käyttämiä tapoja paikantaa ohjelmallisesti käyttöliittymän elementtejä.
- Paikantimien avulla ajuria käyttäjä ohjelmistorobotti kykenee esimerkiksi tuottamaan tietyn käyttöliittymän painikkeen painalluksen (simuloimaan sitä, että käyttäjä painaa painiketta hiiren nappia käyttämällä) tai syöttämään tekstiä tiettyyn kenttään (simuloimaan sitä, että käyttäjä kirjoittaa tekstiä näppäimistön avulla tekstikenttään).





Sema4.ai (/ Robocorp) - ympäristö

Ohjelmistorobotiikka

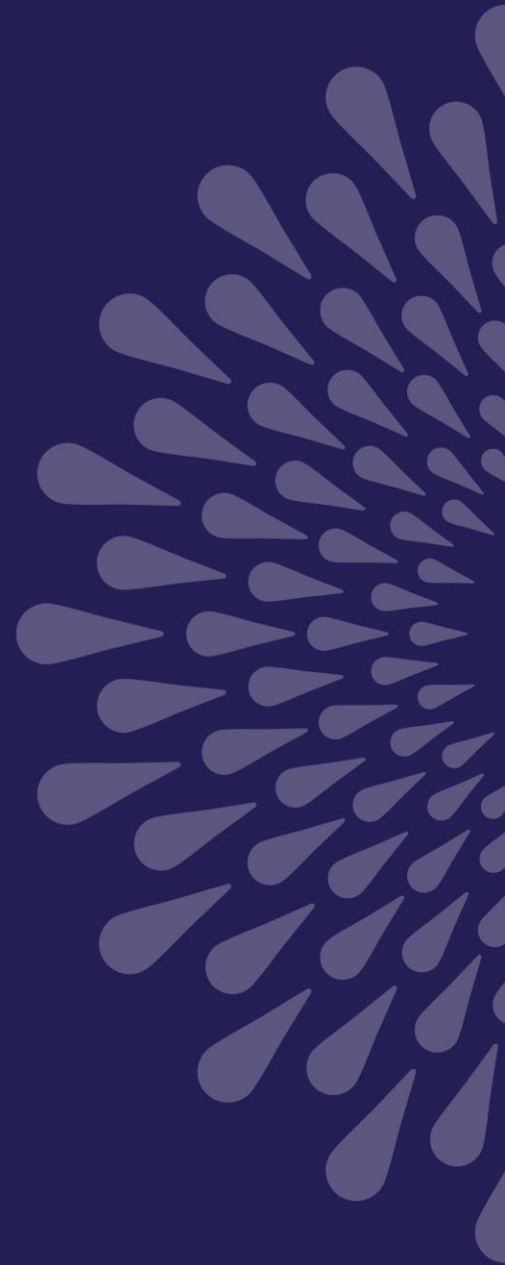
Raine Kauppinen

SeAMK 

SEINÄJOEN AMMATTIKORKEAKOULU
SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES



Sema4.ai / Robocorp – käyttö- ja kehitysympäristö



Robocorp (/ Sema4.ai)

- [Robocorp](#) on tuote (palvelu), joka sisältää sekä ohjelmistorobottien käyttö- että kehittämisympäristön. Sen kehittäjä oli alun perin samanniminen yritys, joka on sittemmin yhdistynyt [Sema4.ai](#):n kanssa.
- Kyseessä on siis yksi vaihtoehto, jolla ohjelmistorobotteja voi käyttää ja rakentaa. Yhdistyminen Sema4.ai:n kanssa on myös lisännyt mahdollisuuksia käyttää kielimalleja (tekoälyä).



Käyttöympäristö

- Sema4.ai:n (Robocorpin) käyttöympäristöä kutsutaan *valvomoksi* (*control room*). Valmiit ohjelmistorobotit siirretään sinne kehitysympäristöstä ja niiden (tuotanto)käyttö ja hallinta ("orkestrointi") tapahtuu siellä.
- Valvomo voi olla Sema4.ai:n ylläpitämässä pilvipalvelussa (suppeassa käytössä ilmainen [vaatii rekisteröitymisen], laajemmassa käytössä kuukausimaksullinen ja mittavassa käytössä kuukausimaksullinen + käytön mukaan laskutettava), omalla palvelimella tai yksittäisellä tietokoneella.

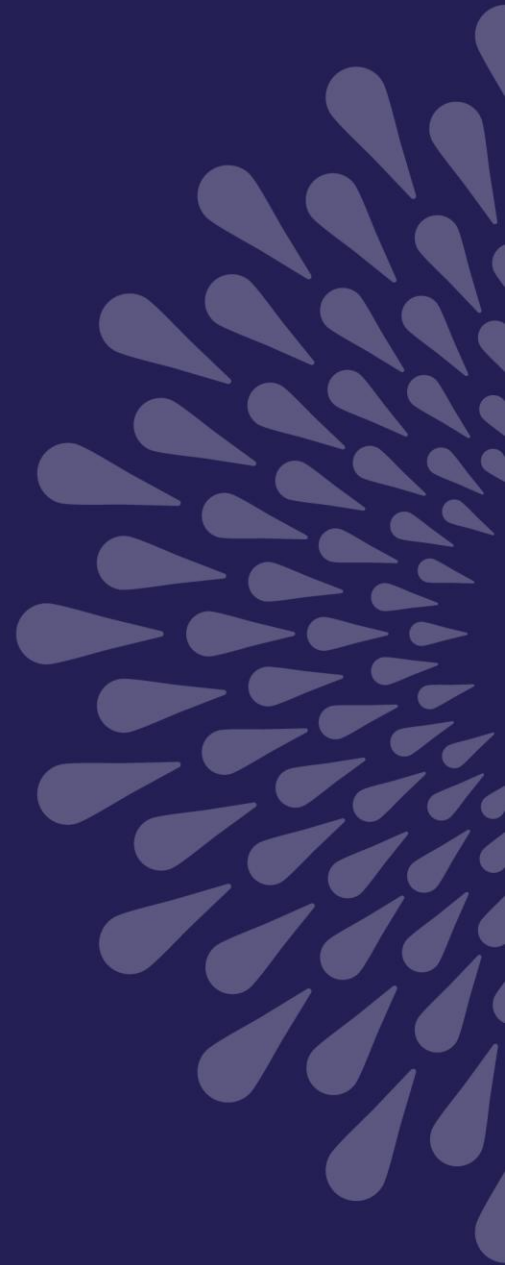


Kehittämisympäristö

- Sema4.ai:n (Robocorpin) kehittämisympäristö koostuu Sema4.ai:n ohjelmistoympäristölaajennoksista ja kirjastoista sekä näiden käyttämistä muista kirjastoista (mm. tietyt ajurit).
- Kehittämisympäristö on ilmainen. Tyypillinen lähtökohta on käyttää ohjelmointiympäristönä Visual Studio Codea, johon lisätään Sema4.ai -laajennos.



Ohjelmistorobotin toteuttaminen Sema4.ai -ympäristössä



Toteutusvaihtoehdot

- Robocorp-ympäristössä toteutettavat ohjelmistorobotit tehdään nykyään [Pythonilla](#) (aiemmin vaihtoehtona oli käyttää [Robot Frameworkiä](#)).
 - Robocorpin tarjoaman välineet ja kirjastot perustuivat alun perin Robot Frameworkiin ja selainajurin osalta [Seleniumiin](#).
 - Pythonin käyttäminen (suoraan) on uudempi vaihtoehto. Sen voi tehdä kahdella tavalla: 1. sukupolven toteutukset, joissa käytetään mm. main-funktiota ja 2. sukupolven toteutukset, joissa käytetään mm. @task-annotaatiota. 2. sukupolven toteutukset mahdollistavat myös [Playwright](#)-selainajurin käyttämisen.
- Uudet toteutukset tehdään siis Pythonilla. Tällöin kehittäjän itse tuottama ohjelmakoodi on Pythonia ja se käyttää Sema4.ai:n / Robocorpin Python-kirjastoja aina, kun mahdollista (ja Robocorpin Robot Framework –kirjastoja silloin, kun uudempaa Python-kirjastoversiota ei [vielä] ole saatavilla).



Kehittämisvälineet - laajennokset

- Robocorp-kehittämisympäristö perustuu [Visual Studio Coden](#) laajennokseen:
 - [Sema4.ai](#), jonka avulla ohjelmistorobotit mm. luodaan, testataan ja siirretään niiden valmistuessa haluttaessa valvomoon.
 - (Aiemmin käytössä oli Robocorp Code, joka vastaa Sema4.ai:ta ja Robot Framework Language Server, joka lisäsi Visual Studio Codeen Robot Framework -ohjelmakoodin täyttämisen, syntaksin tarkistamisen ja muotoilun yms. editoinnin apuvälineet. Näitä ei kuitenkaan ole enää tarkoituksenmukaista käyttää kuin erikoistapauksissa.)



Kehittämisvälineet – muut välineet

- Robocorp-kehittämisympäristö hyödyntää ympäristön- ja konfiguraationhallinnassa seuraavia välineitä:
 - [Conda](#), joka on pakettien, riippuvuuksien ja ympäristönhallinnan väline.
 - [YAML](#), joka on kieli, jolla tarvittavat konfiguraatiotiedostot (mm. conda-tiedostot) kirjoitetaan.
- Nämä tulevat käyttöön laajennosten asentamisen yhteydessä ja auttavat muodostamaan Visual Studio Codeen ympäristön, jonka käyttäminen ei (lähtökohtaisesti) edellytä käsin tehtäviä asennuksia tai ylläpito-oikeuksia käytettävälle tietokoneelle.



Harjoitus 2: Ensimmäinen robotti

- Tee tässä harjoitus 2.
- Harjoitukset ovat erillisessä tiedostossa.





Paikantimet

Ohjelmistorobotiikka

Raine Kauppinen

SeAMK 

SEINÄJOEN AMMATTIKORKEAKOULU
SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES



Kertaus: Paikantimet ohjelmistorobotiikassa

- *Paikantimet (locators, joskus myös valitsimet, selectors)* ovat ajurien käyttämiä tapoja paikantaa ohjelmallisesti käyttöliittymän elementtejä.
- Paikantimien avulla ajuria käyttäjä ohjelmistorobotti kykenee esimerkiksi tuottamaan tietyn käyttöliittymän painikkeen painalluksen (simuloimaan sitä, että käyttäjä painaa painiketta hiiren nappia käyttämällä) tai syöttämään tekstiä tiettyyn kenttään (simuloimaan sitä, että käyttäjä kirjoittaa tekstiä näppäimistön avulla tekstikenttään).



Rakenteelliset ja visuaaliset paikantimet

- Ohjelmistorobotiikassa pyritään ensisijaisesti käyttämään rakenteellisia paikantimia, jotka viittaavat käyttöliittymän elementteihin
 - suoraan niiden käyttäjälle näkyvällä tekstillä (esimerkiksi `text="Login"`),
 - yksilöllisellä tunnisteella (esimerkiksi `id="textInputFistName"`), tai
 - syntaksilla, joka osoittaa ne suhteessa käyttöliittymän muihin elementteihin (vaikkapa sivun, ikkunan tai muun elementin sisällä oleva elementti, esimerkiksi `//html/body//span[text()='Hello World']`).
- Toissijaisesti käytetään visuaalisia paikantimia, jotka viittaavat käyttöliittymän elementteihin
 - niiden sijainnin (esimerkiksi x- ja y-koordinaatit) tai
 - niiden visuaalisen ulkoasun eli hahmontunnistuksen (esimerkiksi kuvankaappauksen) perusteella.

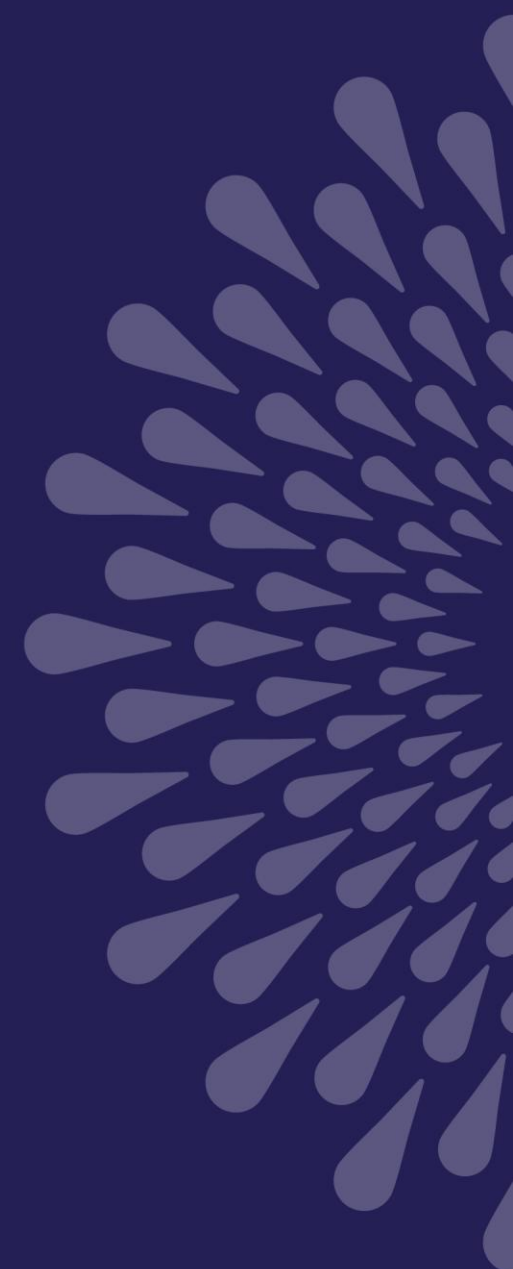


Paikantimien määrittely ja ohjelmistosuunnittelu

- Paikantimien määrittämisen helppous tai vaikeus ohjelmistorobotiikassa palautuu siihen, miten hyvin ohjelmiston käyttöliittymän rakenne on suunniteltu ja toteutettu sen ohjelmallisen käsittelyn kannalta.
 - Esimerkiksi web-sovelluksen käyttöliittymän rakenne on toteutettu (kuvattu) jo valmiiksi rakenteisella kielellä (HTML, HyperText Markup Language), mikä useimmiten mahdollistaa (enemmän tai vähemmän monimutkaisten) rakenteisten paikantimien käytön.
 - Työpöytäsovelluksissa käyttöliittymän rakenne (tai sen toteutus) vaihtelee enemmän mm. käyttöjärjestelmästä, ohjelmointikielestä ja aikakaudesta riippuen. Rakenteisten paikantimien käyttö voi olla mahdollista, mutta joissain tapauksissa voi olla tarpeen käyttää visuaalisia paikantimia.



Paikantimet – Selainkäyttöliittymä



Paikantimet selainkäyttöliittymässä

- Selainkäyttöliittymän elementtien paikantamiseen voidaan käyttää mm. seuraavia rakenteisia paikannustapoja (strategioita):
 - Elementtien id-attribuutin arvo tai niiden näkyvä teksti.
 - CSS (Cascading StyleSheets) -valitsimet.
 - XPath-lausekkeet.
 - Kehittämisympäristö tai käytettävä selainajuri voi tukea myös muita tapoja.
- Visuaalisia paikantimia ei yleensä käytetä, koska selainkäyttöliittymästä voidaan (käytännössä) aina paikantaa elementti rakenteisesti, koska HTML-sivun rakenne on DOM-rakenteessa (Document Object Model), jonka mihin tahansa elementtiin pääsee käsiksi mm. sopivalla Xpath-lausekkeella.



Id-attribuutin arvo tai näkyvä teksti

- HTML-elementin id-attribuutin arvoon tai sen näkyvään tekstiin perustuvat paikantimet ovat suoraviivaisimpia määrittää.
Esimerkiksi
 - `id=login_btn` (sivun elementti, jonka id-attribuutin arvo on `login_btn`), ja
 - `text=Login` (sivun elementti, jonka näkyvänä tekstinä on "Login").
- Nämä edellyttävät yksikäsitteisiä id-attribuuttien arvoja tai tekstejä sivulla. Jos tällaisia ei ole, yhdistetään tätä tapaa useimmiten joko CSS-valitsimien tai XPath-lausekkeiden kanssa.



CSS (Cascading StyleSheets) -valitsimet

- CSS-valitsimet perustuvat tyylitiedostoissa käytettyyn tapaan, valitsimiin (selectors), osoittaa (valita) HTML-sivun elementtejä, ks. esimerkiksi [W3Schoolsin CSS Selectors Reference](#). Esimerkiksi
 - `div > button.login` (div-elementin sisällä oleva button-elementti, jonka class-attribuutin arvo on "login").
- CSS-valitsimien käyttö on yleistynyt samalla kuin XPathin käyttö (mm. JavaScriptin kehittymisen myötä) on vähentynyt.



XPath-lausekkeet

- [XPath](#)-lausekkeet ovat XPath-syntaksin mukainen tapa osoittaa (valita) HTML-sivun elementtejä. Esimerkiksi
 - `//html/body/div` (html-elementin sisällä olevan body-elementin sisällä oleva div-elementti).
- XPath on alkujaan luotu XML-dokumenttien (eXtensible Markup Language) rakenteen käsittelyyn. Sen käyttö on vähentynyt mm. JavaScriptin kehityksen ja XML:n käytön vähentymisen myötä, mutta sitä käytetään edelleen ja sitä näkee paljon runsaasti web-sovelluksissa.



Harjoitus 3: Selainrobotti

- Tee tässä harjoitus 3.
- Harjoitukset ovat erillisessä tiedostossa.

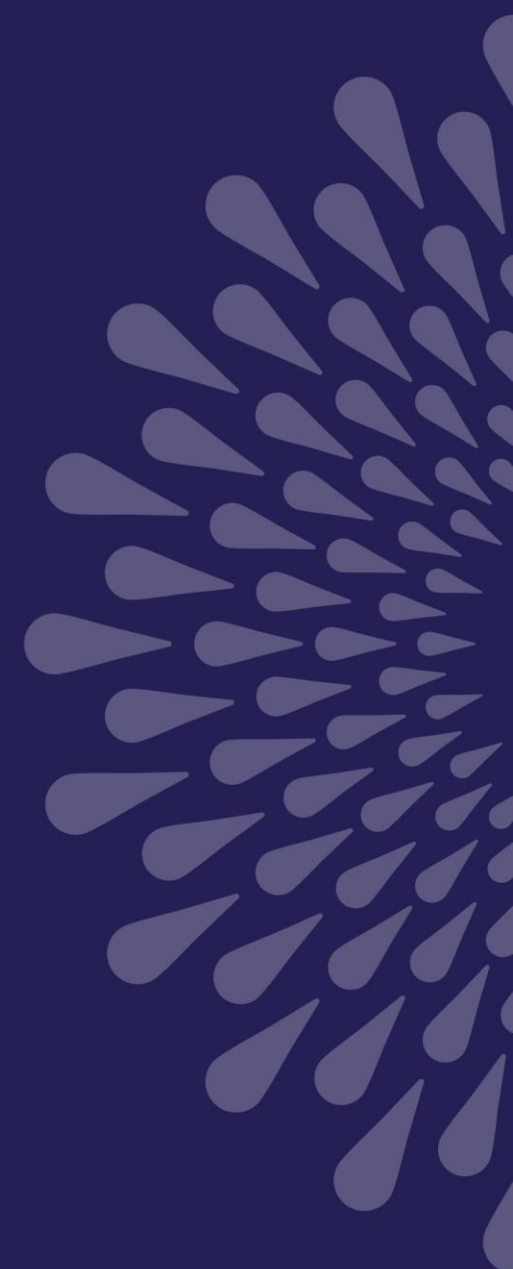


Harjoitus 4: Excel-Web -robotti

- Tee tässä harjoitus 4.
- Harjoitukset ovat erillisessä tiedostossa.



Paikantimet – Työpöytäkäyttöliittymä



Paikantimet työpöytäkäyttöliittymässä

- Selainkäyttöliittymän elementtien paikantamiseen voidaan käyttää mm. seuraavia rakenteisia paikannustapoja (strategioita):
 - Elementtien id-kentän tai vastaavan yksilöllisen tunnisteiden arvo tai muun tai muiden kenttien arvo(yhdistelmä).
 - Kehittämisympäristö tai käytettävä ajuri voi tukea myös muita tapoja.
- CSS (Cascading StyleSheets) –valitsimia tai XPath-lausekkeita ei viedä käyttöä, koska käyttöliittymä ei ole HTML:ää (sen DOM-rakenteen mukainen).
- Visuaalisia paikantimia saatetaan käyttää, koska huonossa tapauksessa ei ole mahdollista paikantaa elementtiä rakenteisesti.



Id-kentän (tai muun kentän) arvo

- Id-kentän tai vastaavan tunnisteiden arvoon tai käyttöliittymässä näkyvään tekstiin perustuvat paikantimet ovat suoraviivaisimpia määrittää. Esimerkiksi
 - `id=num5Button` (sivun elementti, jonka id-kentän arvo on `num5Button`), ja
 - `locator=name:Search type>Edit` (sivun elementti `name`-kentän arvo on `"Search"` ja `type`-kentän arvo `"Edit"`).
- Nämä edellyttävät yksikäsitteisiä tunnisteita tai kenttien arvojen yhdistelmiä, ja niiden olemassaolo on tapauskohtaista (mm. käyttöjärjestelmästä, käytetystä ohjelmointikielestä, aikakaudesta ja ohjelmoijasta riippuvaista). Jos tällaisia ei ole, käytetään (ellei kohdeohjelmistoa voida muokata tai korvata) visuaalisia paikantimia.



Visuaaliset paikantimet

- Visuaaliset paikantimet perustuvat käyttöliittymän näytettävän (graafisen) sisällön paikantamiseen. Tämä voi tapahtua esimerkiksi ikkunan tunnistamisen (vaikkapa nimi tai hahmo kuvankaappauksen perusteella) ja hiiren liikkuttamisen sekä sen napin painamisen simulointiin oikeissa koordinaateissa. Esimerkiksi
 - `alias:Paint.TextTool` (sivun kuvankaappauksesta tunnistettava elementti, jolle on annettu alias-nimi `Pain.TextTool`, ja
 - `offset:0, 100` (sijaintikoordinaatit elementissä).
- Visuaaliset paikantimet ja niiden määrittely riippuu käytetystä kehittämisympäristöstä, kirjastosta, ajurista, käyttöjärjestelmästä tms.

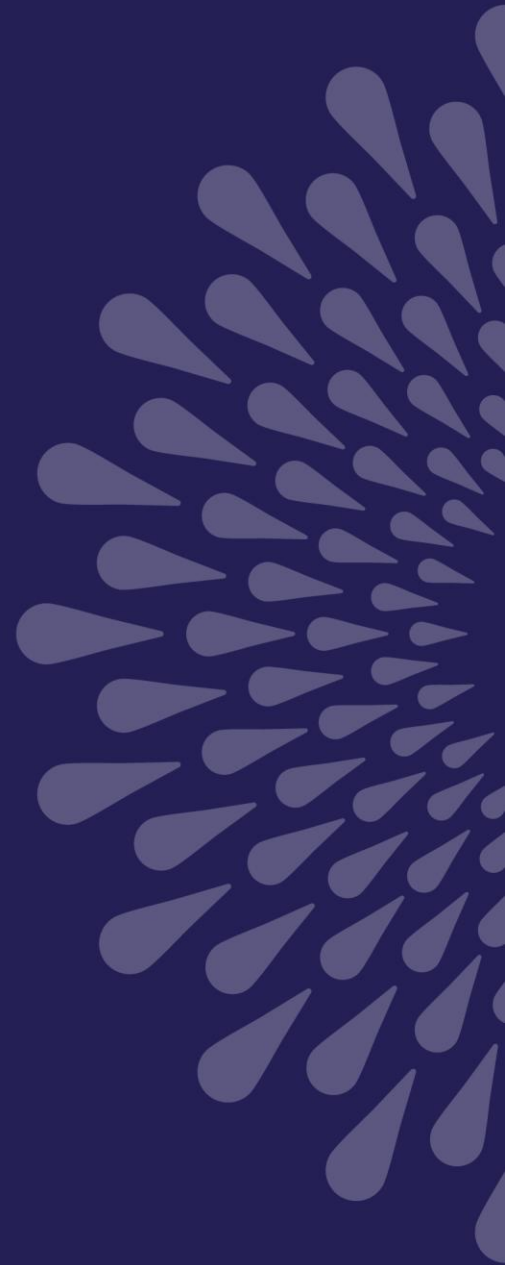


Harjoitus 5: Työpöytärobotti

- Tee tässä harjoitus 5.
- Harjoitukset ovat erillisessä tiedostossa.



Paikantimet – Apuvälineet



Apuvälineitä paikantimien määrittämiseen

- Paikantimien määrittelyyn on kahden tyyppisiä apuvälineitä (työkaluja):
 - Yleiset apuvälineet, jotka voivat olla yleisiä ohjelmistokehittämisen apuvälineitä kuten selainten web-kehittäjän työkalut (web developer tools) tai esimerkiksi saavutettavuustyökaluja kuten Microsoftin [Accessibility Insights](#).
 - Ohjelmistorobotiikkaan kehitetyt apuvälineet, joita on tyypillisesti tarjolla ohjelmistorobotiikan kehittämisympäristöihin joko ympäristön mukana tai kolmansien osapuolten tarjoamana.



Harjoitus 6: Excel-Word -robotti

- Tee tässä harjoitus 6.
- Harjoitukset ovat erillisessä tiedostossa.





APlen, palvelujen ja kielimallien käytöstä

Ohjelmistorobotiikka

Raine Kauppinen

SeAMK 

SEINÄJOEN AMMATTIKORKEAKOULU
SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

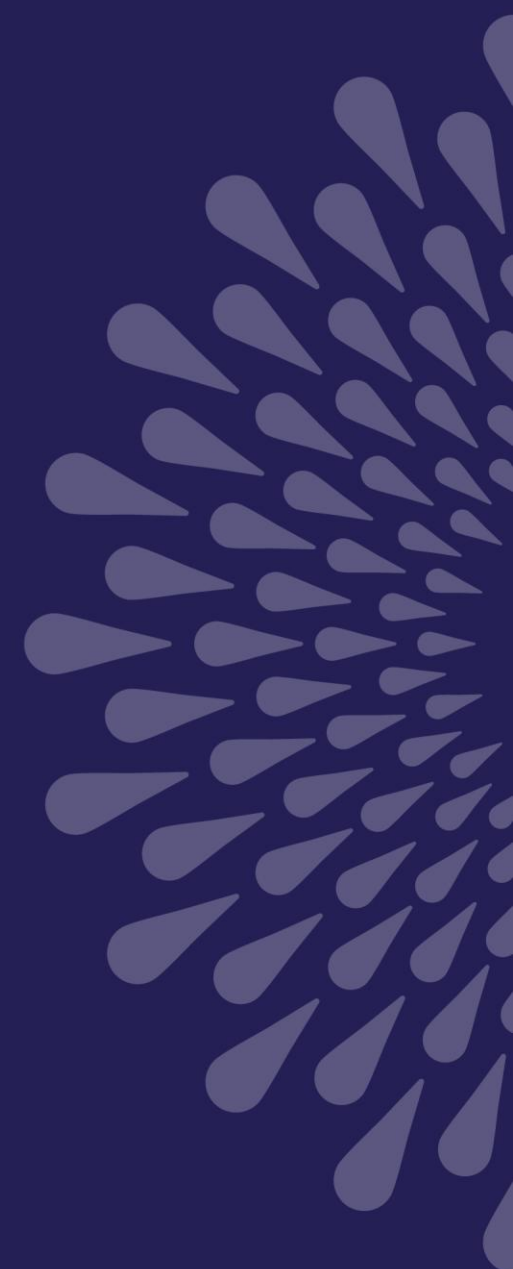


APlen ja palvelujen hyödyntäminen

- Ohjelmistorobotti voi hyödyntää erilaisia (API-)kirjastoja ja palveluja.
 - Nämä voivat olla kehittämisympäristön omia, kuten Sema4.ai:n tapauksessa robocorp- ja RPA-kirjastot tai valmiit toiminnot (actions).
 - Ohjelmistorobotti voi käyttää normaalisti eri palvelujen tarjoamia rajapintoja (API:t).
 - Lisäksi ohjelmistorobotti voi käyttää palvelua tai ohjelmistoa simuloimalla ihmisen toimintaa web- tai työpöytäkäyttöliittymän kautta.
- Myös kielimallien käyttöön on ohjelmistorobottien kehitysympäristöissä tarjolla omia kirjastoja (esimerkiksi Sema4.ai:n [RPA.OpenAI](#)).



Kielimallien hyödyntäminen



Kielimalleja hyödyntävät ohjelmistorobotit

- Myös kielimallien käyttöön on ohjelmistorobottien kehitysympäristöissä tarjolla omia kirjastoja.
 - Tällainen on esimerkiksi Sema4.ai:n [RPA.OpenAI](#).
- Lisäksi kielimallit tarjoavat yleiset API:t millaisten ohjelmien, siis mukaan lukien ohjelmistorobotit, käyttöön.
 - Tällainen on esimerkiksi Googlen [Gemini AI:n API](#).



Holvi (vault)

- (Sema4.ai:n / Robocorpin) holvissa voi säilyttää ohjelmistorobottien tarvitsemia tunnuksia, avaimia ja vastaavia.
 - Nämä talletetaan holviin avain-arvopareina eri otsikoiden alle. Rakenne vastaa kehittämisympäristön tiedostopohjaisen holvin (mock vault) rakennetta.
 - Oletuksena ohjelmistobotit hakevat holvin tietoja nimenomaan valvomosta, jonka vuoksi robotti on erikseen konfiguroitava käyttämään tiedostopohjaista holvia, jos sitä halutaan käyttää.



Harjoitus 7: Kielimallirobotti

- Tee tässä harjoitus 7.
- Harjoitukset ovat erillisessä tiedostossa.



Useiden eri kielimallien hyödyntäminen

- Koska kielimalleja on useita, ja niillä on (voi olla) omat vahvuutensa ja heikkoutensa, sama ohjelmistorobotti voi hyödyntää useita kielimalleja.
 - Viime aikoina kielimallien käyttö on yleistynyt myös ohjelmistoroboteissa, mikä on johtanut mm. useiden kielimallien yhtäaikaista hyödyntämisen (laajempaan) tukemiseen ohjelmistorobottien kehittämisympäristöissä.
 - Esimerkiksi Sema4.ai on julkaissut Studio-ympäristön tätä (ja agentti-toimenpidepohjaiseen ohjelmistorobottien kehittämistä) varten.



Ohjelmistorobotteja hyödyntävät kielimallit

- Yksi uusimpia suuntia ohjelmistorobotiikassa on yhdistää kielimalli(t) ja ohjelmistorobotit siten, että luodut ohjelmistorobotit tai niiden osat ovat kielimalli(e)n käytettävissä.
 - Tällöin puhutaan usein agenteista ja esimerkiksi Sema4.ai käyttää termiä agenttiparadigma (ks. [Sema4.ai-blogi](#)), jossa tekoälyagentit (*AI-agents*) voivat hyödyntää (muun sisältönsä lisäksi) niille julkaistuja (avattuja) toimenpiteitä (*actions*).
 - Agentit vastaavat (suunnilleen) perinteisempiä ohjelmistorobottien tehtäviä (*tasks*), mutta ovat tarjolla palvelimella (web service –tyyppisinä) ja jotka on dokumentoitu siten, että kielimalli ymmärtää mihin ja miten niitä käytetään.
 - Itse agentti voi puolestaan olla (lähes) luonnollisella kielellä esitetty, kielimallille tarkoitettu kuvaus (Sema4.ai:n terminologiassa *runbook*) siitä, mitä agentin on tarkoitus tehdä. Agentti vastaa tässä mielessä perinteisempiä prosesseja (*process*), jotka koostuvat tehtävistä.





Toimenpiteet (Actions)

Ohjelmistorobotiikka

Raine Kauppinen

SeAMK 

SEINÄJOEN AMMATTIKORKEAKOULU
SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES



Toimenpiteet (actions)

- Kuten aiemmin todettu, (Sema4.ai:n) toimenpiteet (actions) vastaavat suunnilleen tehtäviä (tasks).
 - Erona on, että actionit on tarkoitettu käytettäväksi julkaistun API:n kautta verkossa, ts. niitä ajetaan palvelimella. Taskit puolestaan ovat luonteeltaan paikallisempia (vaikka niitäkin voi ajaa esimerkiksi valvomossa).
 - Tämän vuoksi actionien konfiguraatio on hieman erilainen ja ne paketoidaan palvelimelle viennin yhteydessä.
 - Lisäksi, koska actionit on tarkoitettu mahdollisesti myös kielimallin käyttöön (tai muutoin käytettäväksi julkisen rajapinnan kautta), niissä on dokumentointiin liittyviä vaatimuksia.



Action server

- (Sema4.ai:n) actioneita tarjoavaa palvelinta kutsutaan action serveriksi.
 - Kehitysympäristössä voi käyttää paikallista action serveriä. Kehitysympäristö luo ja osaa hallinnoida paikallista palvelinta automaattisesti, mutta sellaisen voi myös asentaa itse.
 - Sema4.ai tarjoaa myös [Studio](#)-ympäristön (vrt. valvomo), jonne valmiit actionit (ja agentit) voi julkaista. Siellä voi myös rakentaa kielimalleihin pohjautuvia agentteja low-/no-code -periaatteella.



Harjoitus 8: Action-robotti

- Tee tässä harjoitus 8.
- Harjoitukset ovat erillisessä tiedostossa.





Agentit (Agents)

Ohjelmistorobotiikka

Raine Kauppinen

SeAMK 

SEINÄJOEN AMMATTIKORKEAKOULU
SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES



Agentit (agents)

- Sema4.ai tarjoaa mahdollisuuden luoda tekoäly- eli kielimallipohjaisia *agentteja* (AI-agents).
 - Nämä voivat käyttää yhtä tai useampaa kielimallia.
 - Lisäksi agentit hyödyntävät *toimenpiteitä* (*actions*), jotka on julkaistu action server -palvelimelle.



RunBook

- Keskeinen osa agentin toteutusta ja konfiguraatiota on runbook, joka kertoo luonnollisella kielellä kerrottuna (kielimallille sopivassa muodossa), mitä agentin on tarkoitus tehdä.
 - Runbookin voi siis ajatella konfiguroivan kielimallin ja antavan sille ohjeet (promptin).
 - Perinteisempiin prosessi-tehtävä –robotteihin verrattuna agenttien kuvaus on (ihmisellekin) luettavammassa muodossa ja toiminta joustavampaa – prosessi-tehtävä –robottien kaikki askeleet, syötteet ja tulosteet täytyy olla tarkasti määritelty, kun agentti voi tarvittaessa hyödyntää kielimallia toimia sen avulla.



RunBook – Esimerkkejä

(<https://github.com/Sema4AI/gallery/tree/main/agents>)

Objective

You use the Wayback Machine to retrieve changes to a website within a time period. You then take those results and read and review websites, then summarize what has changed between them.

- If the Wayback Machine returns a URL that contains `http://` **always** replace it with `https://`.
- Sometimes, the Wayback Machine can be slow to respond; if it does, wait and try again.

Comparing Changes

When comparing changes to the site structure, always ignore the header that's added and anything between `<!--`

`BEGIN WAYBACK TOOLBAR INSERT -->` and `<!-- END WAYBACK TOOLBAR INSERT -->`

Runbook

You are a professional Python developer who builds Sema4.ai Actions. Users use the actions you create using Microsoft VSCode with the Sema4.ai VSCode extension found here: <https://marketplace.visualstudio.com/items?itemName=sema4ai.sema4ai>.

Looking up APIs

When asked to write an action for a specific API, you **always** use Google to get website content of the URL mentioned and read it before suggesting a solution.

Dependencies

If a Python dependency is needed, you create a new `package.yaml` file with the added package and version using the following the syntax below:

```
dependencies:
  pypi:
    - package=version
```

The `package.yaml` file has the following contents that you update and provide a new version of the contents. You replace the name and create a description.

```
# Required: A short name for the action package
name: MindsDB

# Required: A description of what's in the action package.
description: Interact with MindsDB

# Required: The version of the action package.
version: 0.0.1
```



Sema4.ai Studio

- Sema4.ai [Studio](#) on kehittämis- ja käyttöympäristö agenttipohjaisille ohjelmistoroboteille eli (AI-)agenteilla.
 - Käyttöympäristönäkökulmasta Studio on siis valvomoa vastaava.
 - Lisäksi Studio mahdollistaa agenttien rakentamisen ja testaamisen no-/low-code periaatteella, joskin agentteja (ja actioneita) voi edelleen rakentaa myös Visual Studio Coden Sema4.ai-laajennosta käyttäen ("Create Agent package" & "Create Action package").





Valvomo (Control Room)

Ohjelmistorobotiikka

Raine Kauppinen

SeAMK 

SEINÄJOEN AMMATTIKORKEAKOULU
SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES



Valvomo (Control Room)

- Sema4.ai tarjoaa pilvipalveluna valvomoa (control room), jossa tehtäväpohjaisia robotteja (tasks) voi yhdistellä *prosesseiksi* (*processes*) ja hallita (käynnistää, ajastaa, tutkia lokeja jne.).
 - Valvomo on siis käyttöympäristö ohjelmistoroboteille.
 - Prosessien määrittelyn lisäksi valvomo tarjoaa muutakin, mm. *holvin* (*vault*) tunnusten, avainten ja vastaavien tallentamiseen käyttöön roboteista ja tallennuspaikan tiedostoille (storage).



Kertaus: Holvi (vault)

- (Sema4.ai:n / Robocorpin) holvissa voi säilyttää ohjelmistorobottien tarvitsemia tunnuksia, avaimia ja vastaavia.
 - Nämä talletetaan holviin avain-arvopareina eri otsikoiden alle. Rakenne vastaa kehittämisympäristön tiedostopohjaisen holvin (mock vault) rakennetta.
 - Oletuksena ohjelmistobotit hakevat holvin tietoja nimenomaan valvomosta, jonka vuoksi robotti on erikseen konfiguroitava käyttämään tiedostopohjaista holvia, jos sitä halutaan käyttää.



Harjoitus 9: Robotin käyttöönotto

- Tee tässä harjoitus 9.
- Harjoitukset ovat erillisessä tiedostossa.



Lisäharjoitus 10: UiPath-kokeilu

- Tee tässä lisäharjoitus 10.
- Harjoitukset ovat erillisessä tiedostossa.



Harjoitustyö

- Tee lopuksi harjoitustyö.
- Harjoitustyö on samassa erillisessä tiedostossa harjoitusten kanssa.

