



Tekoälyavusteinen ohjelmistokehitys

Havaintoja, oppeja ja harhapolkuja

Juha-Matti Tirilä, Senior AI Developer
Siili Solutions OYJ

DATOS Kielimallit
17.8.2023

Sisältö

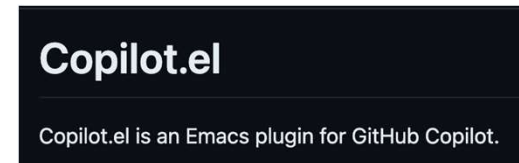
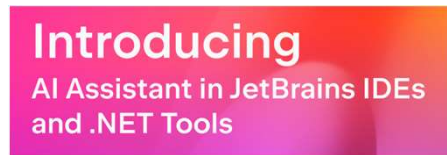
1. Työkaluista (lyhyesti)
2. Ohjelmoinnin uusi aikakausi, eli näin nämä koetaan
3. Kokeiluja, havaintoja
 - Spring Boot –hackathon
 - Amazon Lex Bot & CDK
 - Django & ilmoittautumissovellus
4. Johtopäätökset





Työkaluja

Työkaluja



Visual Studio Code
(+Insiders)

IntelliJ Idea (+ Early Access
Program)

AWS Cloud9 and
CodeWhisperer

Emacs + Plugins



CodeWhisperer not quite getting it (just duplicating the comment)

```
print(response.content)
self.assertContains(response, text: f'<p>IN:</p><ul><li>{self.multigroup_play
self.assertContains(response, text: f'<p>OUT:</p><p>No players yet.</p>', htm
```

Insert Code
→

Previous
←

Next
→

```
previous one, just no optin_buffer,
Suggestion 3 of 3 from CodeWhisperer : ng optin_participant_limit , otherwise similar
# to the previous test.

# Let's add a class similar to the previous one, just no optin_buffer,
# but instead the limiting factor being optin_participant_limit , otherwise similar
```



Työkalut: huomioita

- Tilanne muuttuu nopeasti. “Only available in Developer Preview Version”
- Itse kokeiltua: ChatGPT, Copilot (+X/Chat), JetBrains AI Assistant, VSCode / IntelliJ Idea, CodeWhisperer
- Myös ergonomia elää: pikanäppäimet osin hätäisiä, “toggle next suggestion” etc.
- Vaatii, että ollaan online! (Vrt. Juna Oulu – Seinäjoki)

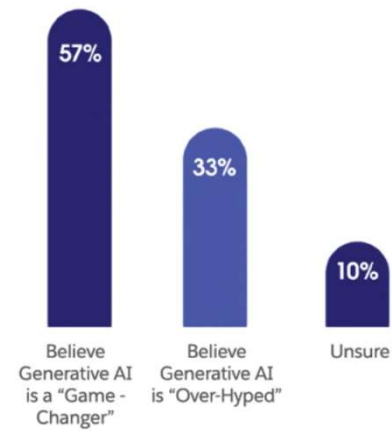
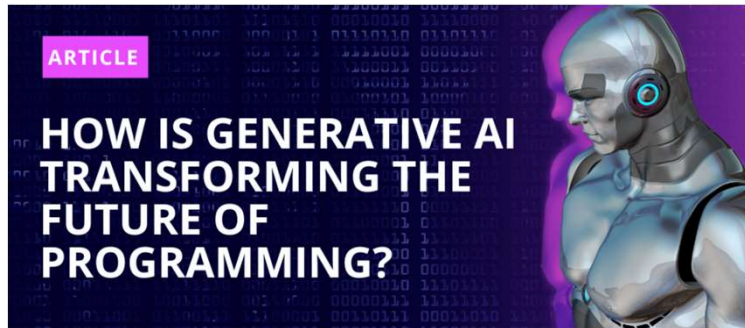




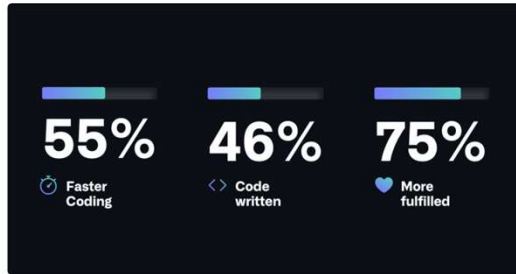
Reaktioita meiltä ja maailmalta



Reaktioita / lähestymistapoja



Reaktioita / lähestymistapoja



Top-level findings that show GitHub Copilot helps developers code faster, completes up to 46% of code, and leaves developers feeling more fulfilled at work.

“It is the same as copy-paste from StackOverflow, just faster and more convenient”

“They are all pretty useless.”

“Programmers who write code with AI helpers make more mistakes, a new study says”



Kokemuksia muutamasta projektista



Hackathon: Spring Boot

- Hae Oulun kaupungin liikenne-rajapinnasta tietoa ja visualisoi jalankulkijoiden määrää eri puolilla kaupunkia
- Teknologiat ennalta tuntemattomia

Spring Boot: off the floor on the board!

JU

I have a skeleton mvn spring boot repository I got from somewhere. Is there some initial setup step I need to perform to e.g. install dependencies, if some have been defined



Assuming that the project is using Maven as the build system, the standard approach for setting up the project and installing the dependencies would be as follows:

1. **Install Java:** First, you need to install Java Development Kit (JDK) as it's a prerequisite for running Spring Boot applications. Spring Boot 2.3 and later supports Java 11.
2. **Install Maven:** Maven is a build automation tool used primarily for Java projects. You should install it if you haven't already.
3. **Check out the project:** If the project is under source control (such as Git), you need to check it out to your local machine.
4. **Go to the project directory:** Navigate to the root directory of the project using a command line. This is typically the directory that contains the `pom.xml` file, which is the Project Object Model file for Maven.
5. **Install dependencies:** Run `mvn clean install` from the command line. This command cleans the target directory of the build (deleting old compiled classes and resources), installs the project dependencies, and build and install the project. Dependencies are usually defined in the `pom.xml` file.
6. **Running the application:** After you've installed the dependencies, you should be able to start the application. If it's a web application, you can usually start it by running `mvn spring-boot:run` from the command line.

Regenerate



Spring boot & Java & ChatGPT Hackathonissa

- Yleisistä linjoista välitön yleiskuva
 - Nollatilanteesta riippuvuuksien asennus ja ylätasen ohjeet, miten projektissa edetään
 - Mutta vertaa kuitenkin ihmisen tuottamaan / validoimaan dokumentaatioon
- Myös koodiehdotukset pääosin toimivia, nopeutti selvästi uudella teknologialla työskentelyä
- Ryhmän konsensus: saatiin tehtyä toteutus, johon ei olisi päästy ilman ChatGPT:tä



POC: Amazon Lex Botit CDK:lla

- Pyrkimys määritellä koko infra ja sovellus AWS CDK:lla
- Pääasiallinen työkalu: ChatGPT

Ei toimi: Lex Botit

I know there is a concept called "output context". I guess these are something that I could use to "save" slot values to be further used for other intents.

Can you tell me how to do this.

You should probably know that I am operating my Bot using CDK, so I don't want to know how to do this from the AWS Management Console.

 Certainly! In Amazon Lex, output contexts allow you to pass information (such as slot values) from one intent to another within a conversation. By using output contexts, you can influence the behavior of subsequent intents based on the information collected earlier. Here's how you can configure output contexts in Amazon Lex using the AWS CDK:

1. First, you need to define your intent that sets the output context. For example:

```
typescript
import * as cdk from 'aws-cdk-lib';
import * as lex from 'aws-cdk-lib/aws-lex';

export class LexBotStack extends cdk.Stack {
  constructor(scope: cdk.Construct, id: string, props?: cdk.StackProps) {
    super(scope, id, props);
  }
}
```

 Copy code

 Regenerate



Lex-botit & CDK: mikä meni pieleen?

Mistä olisi voinut tietää, että on parempi palata perinteisten dokumenttien ääreen?

Alusta lähtien:

- Sekaisin eri Lex-versioihin liittyviä koodinpätkiä
- Vaikeuksia löytää selkeitä luotettavia lähteitä, joissa asia dokumentoitu
 - Jos huonosti ihmisille suunnattua dokumentaatiota tai esimerkkejä, myös ChatGPT:n vaikea arvata oikein
- Myöskään yritykset keskustella ChatGPT:n kanssa ja korjata sen käsityksiä eivät auttaneet

Miksi vaikeutta ei tunnistettu ajoissa

- ChatGPT tuotti sinänsä uskottavan näköisiä ja jopa tyyllillisesti hyviä ehdotuksia
- Aikaisempi kokemus (/toive) siitä, että kontekstomalla paremmin asia saadaan ratkaistua
 - Tulisi tunnistaa tilanne, jossa tarvittavaa tietoa ei vain ole käytettävissä

Ongelmat ratkaistiin lukemalla AWS:n perinteistä dokumentaatiota (lukuisten yritysten ja erehdysten jälkeen).



POC: Django-sovellus IDEen integroitua assistenttia käyttäen

- Toiminnallisuus: vrt. Nimenhuuto.com (++)
- Pääasialliset työkalut:
 - VS Code (insiders)
 - GitHub Copilot Chat

Django-POC: VSCode Chat-ikkunassa



jtirila

In a Django project, can you describe me the general process of adding a new model (database table) and a web forms to create new instances and editing and deleting previously created ones? I'd like both a high level overview and a list of files to create/edit.



GitHub Copilot



Sure, I'd be happy to provide a high-level overview of the process of adding a new model and web forms to a Django project, as well as a list of files to create or edit.

1. Define the Model: The first step is to define the new model in the `models.py` file of the app where you want to add it. You can define the fields of the model using Django's built-in field types, such as `CharField`, `IntegerField`, `DateField`, etc. You can also define relationships between models using fields such as `ForeignKey`, `ManyToManyField`, etc.
2. Create the Database Table: After defining the model, you need to create the database table for it. You can do this by running the following command in the terminal: `python manage.py makemigrations <app_name>`, followed by `python manage.py migrate`.
3. Create the Form: Next, you need to create a form that will allow users to create new instances of the model. You can do this by creating a new file in the `forms.py` file of the app where you want to add the form. You can use Django's built-in form fields, such as `CharField`, `IntegerField`, `DateField`, etc. to create the form fields.



Django-POC: VSCode Chat-ikkunassa



jtirila

OK.

In the Django project that I have, I'd like to start writing some simple tests. Preferably end to end, as in "there is a user and an event, the user clicks 'IN'" --> the users who have opted in for the event should now include this user. How should I go about this?



GitHub Copilot



To write end-to-end tests for your Django project, you can use Django's built-in testing framework.

Here's an example of how you could write a test for the scenario you described:

```
from django.test import TestCase
from django.contrib.auth.models import User
from myapp.models import Event

class EventTestCase(TestCase):
    def setUp(self):
        self.user = User.objects.create_user(username='testuser', password='testpass')
        self.event = Event.objects.create(name='Test Event')

    def test_user_opt_in(self):
        # Log in the user
        self.client.login(username='testuser', password='testpass')
```



Djangon opettelu: miksi toimi paremmin kuin Lex & CDK?

Laajat kysymykset Al:lle:

- Yleiset prosessit
- Vertailut muihin vastaaviin frameworkeihin (Rails)
- Niin Rails kuin Django pitkäaikaisesti stabiileja ratkaisuja, joista > 15 vuoden ajalta hyvää dokumentaatiota netissä

Koodinpätkät & automaattiset ehdotukset

- Lähes aina jotain korjattavaa, mutta silti hyödyllisiä lähtökohtia
- Yllättävän hyvä adaptoituminen kontekstiin
- Helppo verrata todelliseen dokumentaatioon
→ toimimattomien pätkien korjaaminen sujuvaa



Johtopäätökset



Johtopäätöksiä

- “Kun toimii, käytä menemään”
 - Keskeinen haaste: tunnista, milloin avustaja epävarma! Nyhtäminen pääosin ajan tuhlausta
 - Tunnista asiayhteys: jos muuttuvaa / tuoretta teknologiaa, saat useammin ehdotukseksi höpönlöpöä.
- “Käytä silloin, kun helppo tehdä mutta vaikea muistaa ulkoa”, ”boilerplaten generointiin”
- Toimii parhaiten, kun tarjolla on muutenkin paljon laadukasta dokumentaatiota – vertaile ristiin!
 - Tässä tapauksessa auttaa noviisiakin navigoimaan laajoja kokonaisuuksia, pystyy tuottamaan hyödyllisiä tiivistyksiä ja laajoja kaaria, ei pelkkää boilerplatea
- Älä päädy luuppiin, jossa yrität nyhtää assitentilta asteittain parempia vastauksia
 - Olet AI-assistentin kumiankka, roolit vaihtuvat
- Keskeinen haaste: tunnista, milloin avustaja epävarma!
- “Garbage in, garbage out”
 - Ei tarkoita, että Copilot ym. Hyödyttömiä. Tarkoittaa, että laatu vaihtelee ja ihmisen tulee pysyä kuskin paikalla
- Osaamisen luonne muuttuu, enemmän ylätasoa kuva, hyvät kehotteet, kyky edelleen ymmärtää vanhanmallista dokumentaatiota ja vertailla siihen, kyky tunnistaa huonot ehdotukset.





SIILI®