Microsoft

LangChain4J: Use the power of LLMs in Java!

Julien Dubois,
Principal Manager, Java Developer Relations

What is LangChain4J?

- A Java version of LangChain, a JavaScript + Python framework
- Aims to simplify the creation of applications using Large Language Models
- Fully Open Source

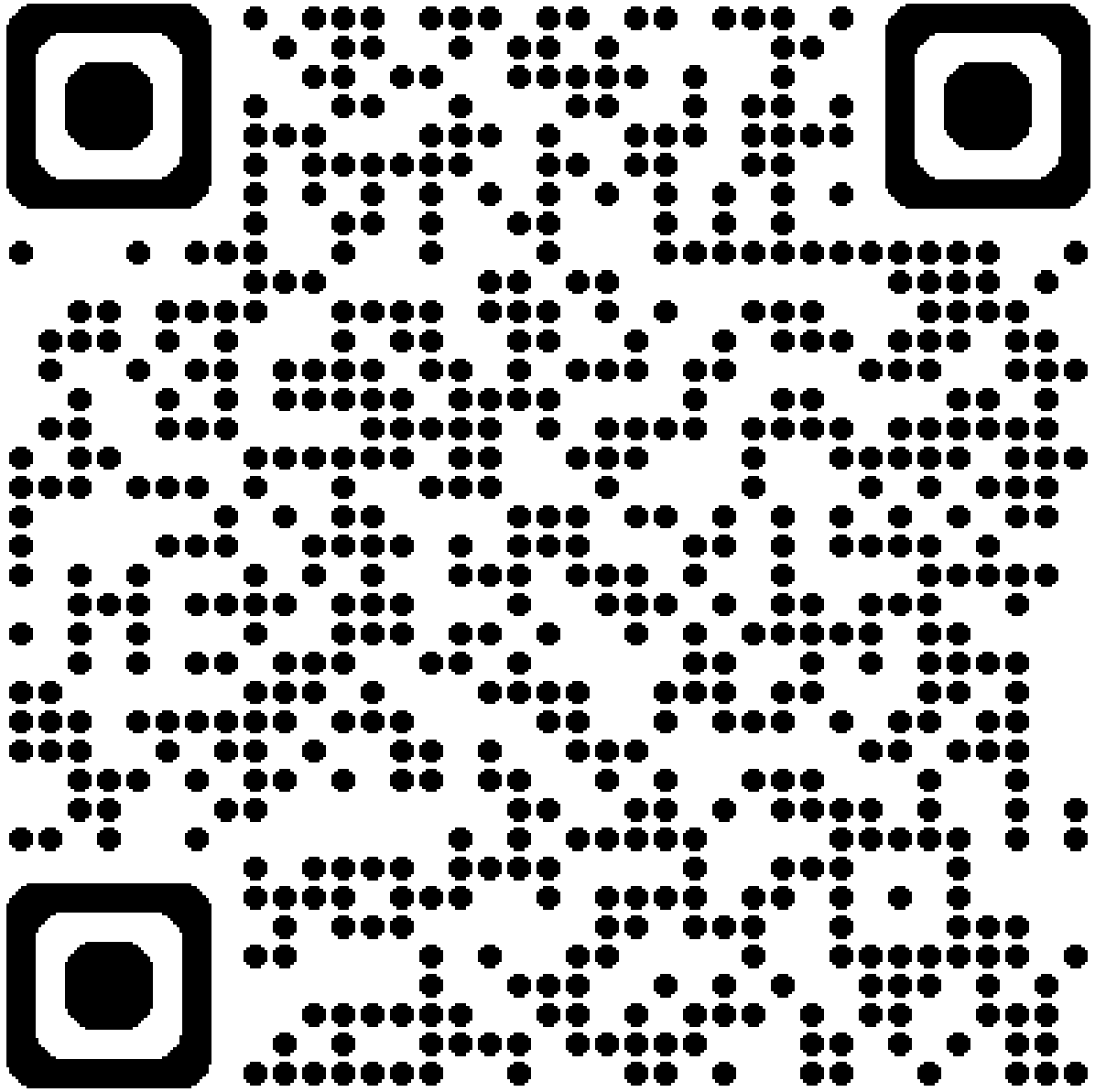
Why would you use LangChain4J?

- A layer of abstraction above many APIs
- Easy integration of those APIs to build more complex applications
- Nice tooling to get work done faster

Demo code

The application used in this demo is available here:

https://github.com/jdubois/jdubois-langchain4j-demo



What we will do

- Basic usage of LangChain4J
  - Image generation
  - Text generation
  - Reasonning
- Commonly used utilities
  - System prompt
  - Memory
- RAG
  - Vector database usage
  - Data ingestion
  - Easy RAG

Running the demos

- **In the cloud**: great for production
  - Advanced models
  - Performance
- **Locally inside Docker**: great for testing
  - Easy to set up
  - No rate limiting
- **Locally on a Mac**: great for developping
  - Much better performance if you have a GPU

RAG = Retrieval-Augmented Generation

- 2 main limits with LLMs
  - They have a cut-off date
  - They don’t have your company internal data
- How do we give more context to an LLM?
  - Ingest new/private data using the same ”embeddings” as the LLM
  - Store that data into a vector database
  - Request the vector database and send that data to the LLM
- This is where LangChain4J excels
  - It allows to easily use vector databases and LLMs
  - It gives you access to advanced features like Hybrid Search and Semantic re-ranking (with Azure AI Search)

Microsoft ❤ developers