

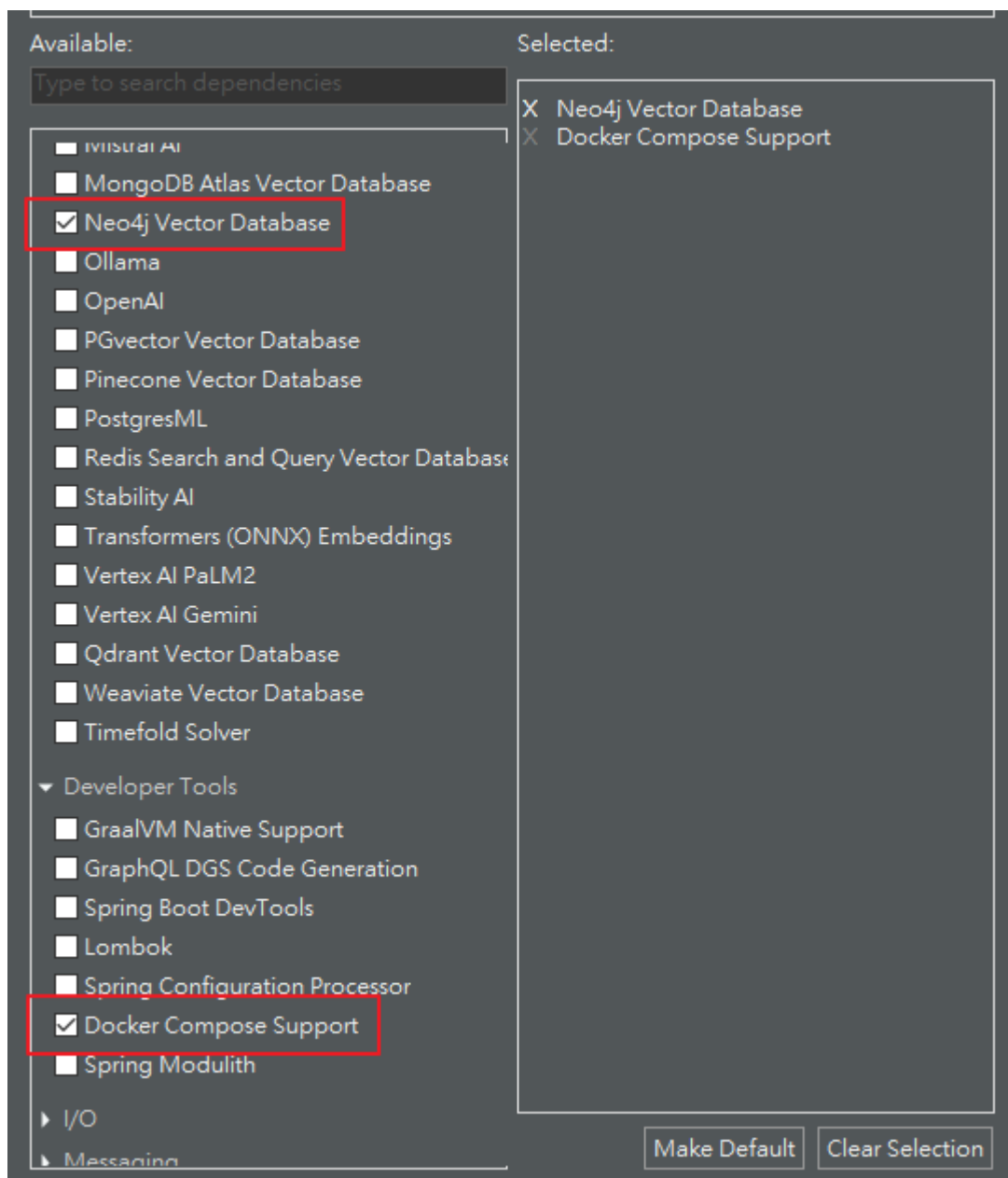
# Day21 - 安裝 neo4j 向量資料庫 (docker)

Day19 有兩個 **Advisor** 還沒說，分別是 **VectorStoreChatMemoryAdvisor** 及 **QuestionAnswerAdvisor**，由於兩者都需要向量資料庫，今天就來教大家透過 **docker** 架設環境

凱文大叔的電腦是 Win 10，安裝 Docker Desktop 是最快速方便的，下面是下載及原廠的安裝教學

- 下載網址: <https://www.docker.com/products/docker-desktop/>
- 安裝教學網址: <https://docs.docker.com/desktop/>

在 Spring Boot 3 以前，我們需要先透過指令方式安裝及啟動 docker 環境，不過 Spring Boot 3 以後多了 Docker Compose Support 套件，只要引入這個 starter 套件，Spring 就能自動跟 docker 整合，另外使用資料庫都還需要驅動程式，可以在 AI 模組下找到 Neo4j Vector Database



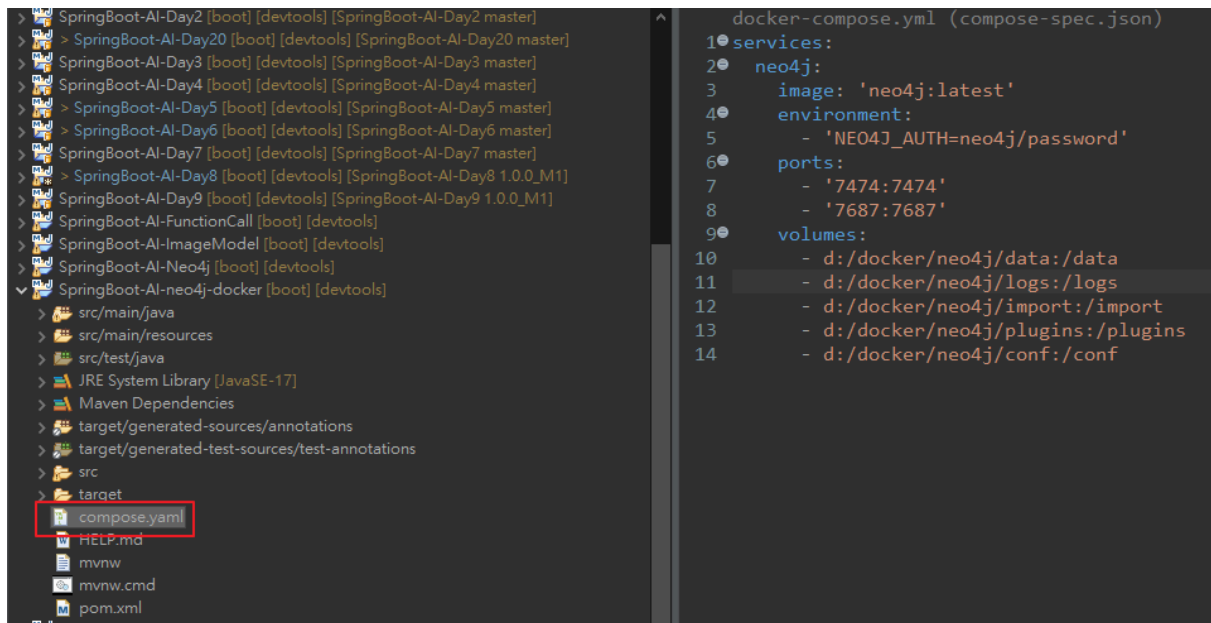
如果已經有專案也可以直接在 pom.xml 加入下面依賴

```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-docker-compose</artifactId>
  <scope>runtime</scope>
  <optional>true</optional>
</dependency>
<dependency>
  <groupId>org.springframework.ai</groupId>
```

```
<artifactId>spring-ai-neo4j-store-spring-boot-starter</ar
</dependency>
```

接著就能設定 docker compose 的 yml 設定檔了，如果建立專案有勾選 Docker Compose Support，這樣專案目錄下就會自動產生一個 compose.yaml，手動修改 pom.xml 就要自己增加了

記得是在專案目錄下，不是 resources 目錄下



支援向量儲存的資料庫非常多，下面這些是 Spring AI 支援的

- [Azure Vector Search](#) - The [Azure](#) vector store.
- [Apache Cassandra](#) - The [Apache Cassandra](#) vector store.
- [Chroma Vector Store](#) - The [Chroma](#) vector store.
- [Elasticsearch Vector Store](#) - The [Elasticsearch](#) vector store.
- [GemFire Vector Store](#) - The [GemFire](#) vector store.
- [Milvus Vector Store](#) - The [Milvus](#) vector store.
- [MongoDB Atlas Vector Store](#) - The [MongoDB Atlas](#) vector store.
- [Neo4j Vector Store](#) - The [Neo4j](#) vector store.
- [PgVectorStore](#) - The [PostgreSQL/PGVector](#) vector store.
- [Pinecone Vector Store](#) - [PineCone](#) vector store.

- Qdrant Vector Store - Qdrant vector store.
- Redis Vector Store - The Redis vector store.
- SAP Hana Vector Store - The SAP HANA vector store.
- Weaviate Vector Store - The Weaviate vector store.

之後大家有興趣也可以測試不同的向量資料庫，程式基本上不用改變，只要更改設定就能切換不同資料庫

至於凱文大叔為何挑選 Neo4j 呢？Neo4j 是一種以 Graph 為資料結構的 NoSQL 資料庫，有沒有覺得很眼熟？沒錯，前幾個月微軟發表的 Graph RAG 正是以 Graph 為資料結構建立 RAG 的知識圖譜，不過鐵人賽凱文大叔只會講到一般的 RAG，未來看有沒有神人將程式搬到 java，我們就可以直接使用了

接著簡單介紹 compose.yaml 的參數

```
services:
  neo4j:
    image: 'neo4j:latest' #docker image, 冒號後是版本, 若要使用最新
    environment:
      - 'NEO4J_AUTH=neo4j/password' #登入Neo4j的帳號密碼
    ports:
      - '7474:7474' #開放的port, 冒號前是對外的port, 冒號後是容器內部
      - '7687:7687' #neo4j 7474 跟 7687都要設定
    volumes:
      #將容器內的檔案儲存在本機端，一來容易修改設定檔，二來方便傳移到其他環境
      #冒號前是本機存放路徑, 冒號後是容器內部路徑
      - d:/docker/neo4j/data:/data
      - d:/docker/neo4j/logs:/logs
      - d:/docker/neo4j/import:/import
      - d:/docker/neo4j/plugins:/plugins
      - d:/docker/neo4j/conf:/conf
```

接著就是見證奇蹟的時刻，啟動 Spring 專案就會看到 log 中顯示 docker 檔案下載的進度條

然後更神奇的事情發生了，凱文大叔並沒有設定 datasource，Spring 竟可以自己建立連線

```

o.s.boot.docker.compose.core.DockerCli : 2cdd9d4bab84 Downloading [=====>] 145.2MB/145.2MB
o.s.boot.docker.compose.core.DockerCli : 2cdd9d4bab84 Downloading [=====>] 145.2MB/145.2MB
o.s.boot.docker.compose.core.DockerCli : 2cdd9d4bab84 Downloading [=====>] 145.2MB/145.2MB
o.s.boot.docker.compose.core.DockerCli : 2cdd9d4bab84 Downloading [=====>] 145.2MB/145.2MB
o.s.boot.docker.compose.core.DockerCli : 2cdd9d4bab84 Downloading [=====>] 145.2MB/145.2MB
o.s.boot.docker.compose.core.DockerCli : 2cdd9d4bab84 Download complete
o.s.boot.docker.compose.core.DockerCli : neo4j Pulled
o.s.boot.docker.compose.core.DockerCli : Network springboot-ai-neo4j-docker_default Creating
o.s.boot.docker.compose.core.DockerCli : Network springboot-ai-neo4j-docker_default Created
o.s.boot.docker.compose.core.DockerCli : Container springboot-ai-neo4j-docker-neo4j-1 Creating
o.s.boot.docker.compose.core.DockerCli : Container springboot-ai-neo4j-docker-neo4j-1 Created
o.s.boot.docker.compose.core.DockerCli : Container springboot-ai-neo4j-docker-neo4j-1 Starting
o.s.boot.docker.compose.core.DockerCli : Container springboot-ai-neo4j-docker-neo4j-1 Started
o.s.boot.docker.compose.core.DockerCli : Container springboot-ai-neo4j-docker-neo4j-1 Waiting
o.s.boot.docker.compose.core.DockerCli : Container springboot-ai-neo4j-docker-neo4j-1 Healthy
o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port 8080 (http)
o.apache.catalina.core.StandardService : Starting service [Tomcat]
o.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/10.1.26]
o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 1002 ms
o.neo4j.driver.internal.DriverFactory : Routing driver instance 140776149 created for server address 127.0.0.1:7687
o.s.b.d.a.OptionalLiveReloadServer : LiveReload server is running on port 35729
o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port 8080 (http) with context path '/'
c.e.a.SpringBootAiNeo4jDockerApplication : Started SpringBootAiNeo4jDockerApplication in 60.68 seconds (process running for 61.47)

```

原來 Spring Docker Compose Support 會自動抓 compose.yaml 的設定作為 datasource，不過這只有測試時比較方便，實際使用還是設定比較好，編輯 application.yml 將 Neo4j 的設定加入吧

```

spring:
  ai:
    openai:
      api-key: ${OPENAI_KEY}
      chat:
        options:
          model: gpt-4o-mini
    neo4j:
      authentication:
        password: password
        username: neo4j
      uri: neo4j+s://localhost

```

明天我們再來透過向量資料庫儲存聊天對話