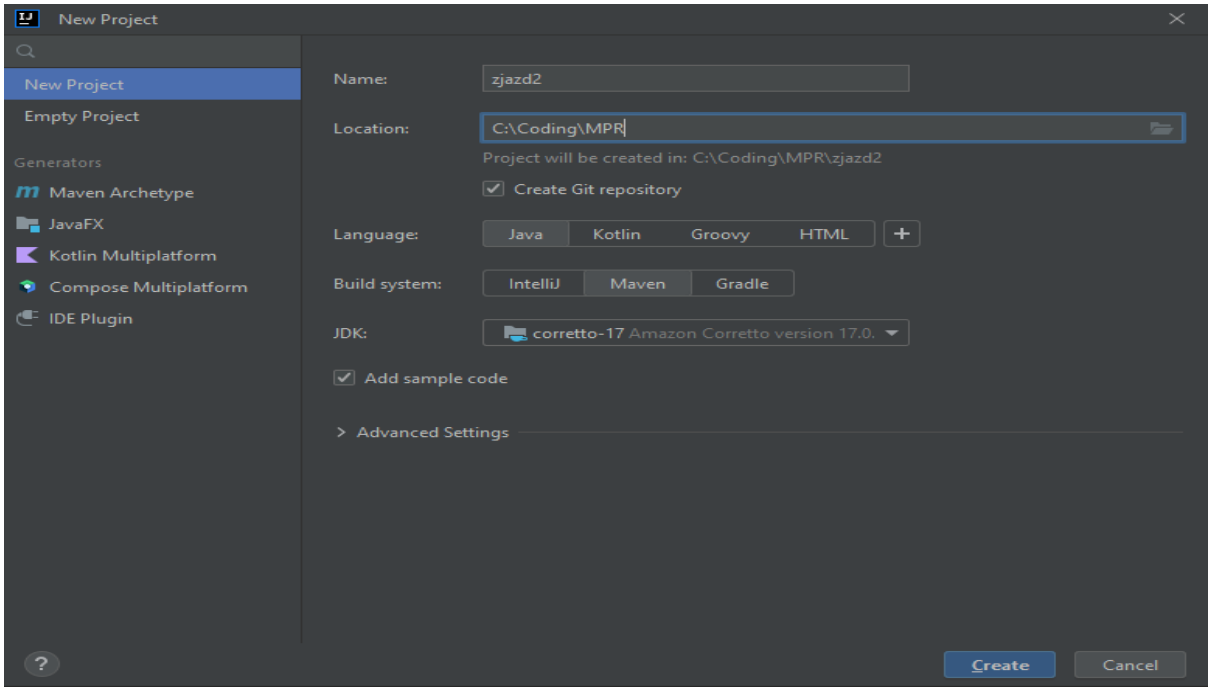
MPR

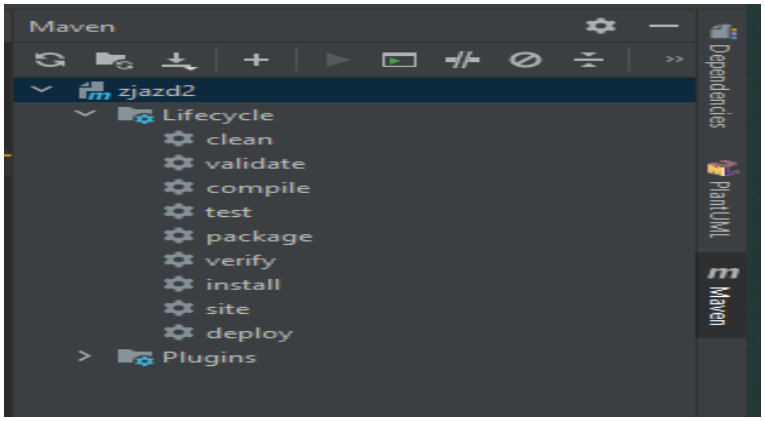
Lab 2

Ćwiczenia zaczniemy od utworzenia nowego projektu z Mavenem



Wybieramy Build system Maven i JDK 17.

Mając to gotowe będziemy mogli korzystając z toolboxa w intellij sterować lifecycle aplikacji.



Powinno to pojawić się po prawej stronie ekranu i po rozwinięciu projektu będziemy mieli takie opcje. By sprawdzić, że wszystko poszło ok klikamy clean a następnie install.

Powinniśmy dostać komunikat o sukcesie a w naszym projekcie pojawi się nowy katalog „target”, w której będzie gotowy JAR z naszą aplikacją.

Zadania

1. W utworzonym już projekcie zaimplementujcie prosty system do obsługi zamówień pizzy. Muszą powstać co najmniej 3 klasy: *PizzaService*, *Pizza* oraz *Order*. Utwórzcie do tego w projekcie katalog „pizza”

   Klasy Pizza i Order będą w tym przypadku naszym modelem i powinny być w osobnym katalogu „model”.

   PizzaService powinna mieć następujące publiczne metody:
   public Order makeOrder(Pizza pizza) -> oczekujemy tutaj zwrócienia informacji o zamówieniu (Numer zamówienia, pizza, cena)
   public List<Pizza> getAvailablePizzas() -> oczekujemy listy dostępnych pizz i ich cen

   To czy potrzebujecie i jakie metody prywatne pozostawiam Wam do decyzji.

   Obsłużcie też przypadek gdy do metody „makeOrder” wpadnie pizza, której nie ma w ofercie. Powinien zostać tutaj rzucony wyjątek (**PizzaNotFoundException**) , który następnie należy złapać w metodzie main. Powinien to być wyjątek „*Unchecked*”

   W metodzie *main* powinniście być w stanie wypisać dostępne pizze a następnie utworzyć 3 zamówienia i każde powinno mieć prawidłowo nadany numer rosnąco zaczynając od 1.

2. Kolejnym krokiem będzie uproszczenie naszych modeli. Aby to zrobić zaimportujcie bibliotekę Lombok korzystając z Mavena. Na stronie https://projectlombok.org/ będzie instrukcja jak prawidłowo dodać do waszego pliku pom.

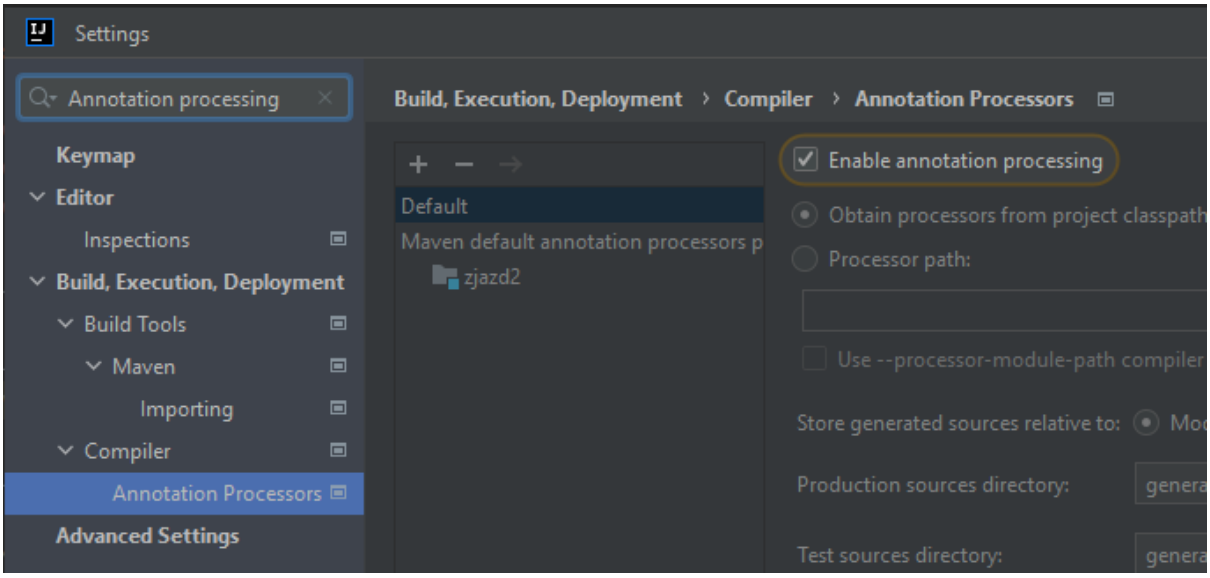
   To co chcemy osiągnąć to by w klasach znajdowały się tylko deklaracje pól, bez konstruktorów ani getterów/seterów.

   Pod tym linkiem macie wypisane możliwości jakie oferuje lombok i wybierzcie odpowiednie z nich.
   https://projectlombok.org/features/

   Adnotacje lomboka można ze sobą łączyć.

   Żeby Lombok prawidłowo pracował trzeba w Intellij włączyć annotation processing.
   Robimy to przez kliknięcie File -> Settings

   Mamy wtedy dostęp do ustawień i możemy w wyszukiwarkę wpisać „Annotation processing”



I należy to włączyć i zapisać.

3. Kolejnym krokiem będzie dodanie obsługi logów. Wykorzystamy do tego kolejną bibliotekę jaką jest log4j.
   Wykorzystamy tutaj dwie zależności:
   https://mvnrepository.com/artifact/org.apache.logging.log4j/log4j-core

   https://mvnrepository.com/artifact/org.apache.logging.log4j/log4j-api

   Wybierzcie dla obydwu te same najaktualniejsze wersje i zaimportujcie do swojego projektu.

   Biblioteka log4j do prawidłowego działania potrzebuje pliku konfiguracyjnego. Nie będziemy teraz zbytnio wchodzić w ten temat jak ktoś jest zainteresowany możliwościami konfiguracji to wszystko będzie w dokumentacji.

   Na dziś wystarczy nam ,że utworzymy w katalogu *resources* nowy plik o nazwie *log4j2.xml* Ważne jest by nazwa była dokładnie taka ponieważ inaczej nasza konfiguracja nie zostanie wczytana.

   Uzupełniamy ten plik przez przekopiowanie tej konfiguracji
   https://goonlinetools.com/snapshot/code/#92x83efiox7io00ybf9a5

   W każdej klasie, w której chcemy skorzystać z loggera musimy go najpierw utworzyć jako statyczny obiekt.

   Przykładowo dla klasy *main* będzie wyglądał on następująco:

   private static final Logger logger = LogManager.getLogger(Main.class);

4. Dodajcie obsługę loggera w klasie *PizzaService* i korzystając z niego wypiszcie komunikaty na kanale info odnośnie tego jaka metoda jest wywołana.
5. W metodzie *main* dodajcie loggera i zalogujcie wyjątek rzucany przy zamówieniu nie istniejącej pizzy z statusem Error i zawartym stacktrace'em błędu.

6. Zadanie dodatkowe: Postarajcie się sami stworzyć identyczny projekt ale tym razem oparty o Gradle. Niech importuje on również lomboka i log4j2. Ułatwi to pracę na kolejnych zajęciach.

Po wykonaniu wszystkich zadań utwórzcie repozytorium na githubie, zapiszcie swój kod oraz wyślijcie linki do mnie na maila jmerta@pjwstk.edu.pl.