## To Do:

- Find a way to increment AnyLogic model time whenever train-function has completed one iteration (or when reset is called). Make sure that the animation is rendered.
- Understand the role of seeds and why despite setting identical seeds outcomes differ between runs → **Done**. Keep in mind that for whatever reason, the seed we set in the configuration of the learning algorithm is not used for the $\varepsilon$-greedy policy (it uses system time instead). Hence randomness cannot be controlled unless the $\varepsilon$-greedy policy is switched off which defeats the point.
- Implement a terminal condition to stop the AnyLogic simulation when train() is finished → **Done**. Implemented two ways of doing this. Either a line in the close() function can be uncommented which will finish the simulation when the training is done. Alternatively, if we do not wish to finish the simulation once training is done (for instance because we wish to load the trained agent), there is also an "exit" button in the main environment now.
- Correct the placement of the cart animation in the main environment → **Done**.
- Save some training output to a log file → **Done**. It turns out that RL4J already automatically creates a log file and adds to it after each training. Implemented a code which deletes the old log file and makes a new one before every run so that each file only contains one training session.
- Implement the loading of the trained policy into a new instance of the MDP and enable it to run for demonstration purposes → **Done** but same problem as with train. The trained agent with the learned policy can be loaded into the environment but as soon as it is told to run, the AnyLogic simulation pauses until the loop is completed.
- Amended most traceln()-calls to display only when the function is called during training (rather than while running the trained agent).
- Once the Cartpole example functions properly, begin work on implementing a different, original learning task (e.g. Callcentre) using the same principle.