UML Diagram

Class Diagram

```
classDiagram

View <|.. StoryFragment
BasePresenter <|-- PagingPresenter~ItemType~
PagingPresenter~ItemType~ <|-- StoryPresenter
StoryFragment "1" <--> "1" StoryPresenter
StoryPresenter "1" --> "1" Cache
BaseService <|-- StatusService
StoryPresenter "1" --> "1" StatusService
StatusService "1" --> "*" GetStoryTask
GetStoryTask "1" --> "1" StatusPageHandler
StatusPageHandler "1" --> "1" PagingServiceObserver~ItemType~
ResultObserver~ResultType~ <|.. PagingServiceObserver~ItemType~
PagingServiceObserver~ItemType~ "*" --> "1" StoryPresenter


class StoryFragment {
    - StoryPresenter presenter
}

class StoryPresenter {
    - StatusService statusService

    + void onScrolled(User)
    # void loadItemsFromService(User, AuthToken)
}

class View {
    <>

    + void setLoadingFooterVisible(Boolean)
    + void displayMessage(String)
    + void cancelMessage()
    + void addItems(List)
}


class PagingPresenter~ItemType~ {
    + PAGE_SIZE$
    - ItemType lastItem
    - boolean hasMorePages
    - boolean isLoading
```

```
    + void loadMoreItems(User user)
    # void loadItemsFromService(User, AuthToken)*
}

class PagingServiceObserver~ItemType~ {
    + void onResultLoaded(Pair, Boolean>)
    + void handleFailure(String)
    + void handleException(Exception)
}

class BasePresenter {
    <>

    # View view
}

class ResultObserver~ResultType~ {
    <>

    + void onResultLoaded(ResultType)*
    + void handleFailure(String)*
    + void handleException(Exception)*
}

class Cache {
    + Cache getInstance()$
    + AuthToken getCurrUserAuthToken()
    + User getCurrUser()
}


class StatusService {
    + void loadStory(AuthToken, User, int, Status,
ResultObserver~Pair~List~Status~, Boolean~~)
}

class BaseService {
    <>

    # void executeTask(Runnable)
}

class GetStoryTask {
    - AuthToken authToken
    - User targetUser
    - Status lastItem
    - int limit
    - Handler messageHandler
```

```
    + public void run()
}

class StatusPageHandler {
    + void handleMessage(Message msg)
    # void handleSuccessMessage(ResultObserver, Boolean>>)
}
```

Sequence Diagram

```
sequenceDiagram

actor User

participant LoginFragment
participant LoginPresenter
participant UserServiceObserver
participant Cache

User ->>+ LoginFragment : onLoginClick()
LoginFragment ->>+ LoginPresenter : onLoginClick(userAlias, password)

LoginPresenter ->>+ LoginPresenter : validateLogin(userAlias, password)
LoginPresenter ->>+ LoginFragment : setErrorText(null)
LoginFragment -->>- LoginPresenter :
LoginPresenter ->>+ LoginFragment : displayMessage("Logging In...")
LoginFragment -->>- LoginPresenter :

LoginPresenter ->>+ UserService : loginUser(userAlias, password,
userServiceObserver)
UserService ->>+ UserService : executeTask(loginTask)
UserService -)+ LoginTask : run()
UserService -->>- LoginPresenter :
deactivate UserService

LoginPresenter -->>- LoginFragment :
deactivate LoginPresenter
LoginFragment -->>- User :

LoginTask ->>+ LoginTask : runTask()
LoginTask ->>+ LoginTask : runAuthenticationTask()
LoginTask -->>- LoginTask : Pair(user, authToken)
LoginTask ->>+ LoginTask : sendSuccessMessage()
LoginTask ->>+ LoginTask : sendMessage(Pair(user, authToken))
LoginTask -)+ UserAuthHandler : sendMessage(Pair(user, authToken))
```

```
  deactivate LoginTask
  deactivate LoginTask
  deactivate LoginTask
  deactivate LoginTask

  UserAuthHandler ->>+ UserAuthHandler : handleMessage(Pair(user, authToken))
  UserAuthHandler ->>+ UserAuthHandler :
  handleSuccessMessage(userServiceObserver, Pair(user, authToken))
  UserAuthHandler ->>+ UserServiceObserver : onResultLoaded(Pair(user,
  authToken))
  deactivate UserAuthHandler
  deactivate UserAuthHandler
  deactivate UserAuthHandler

  UserServiceObserver ->>+ Cache : setCurrUser(user)
  Cache -->>- UserServiceObserver :
  UserServiceObserver ->>+ Cache : setCurrUserAuthToken(authToken)
  Cache -->>- UserServiceObserver :
  UserServiceObserver ->>+ LoginFragment : setCurrentUser(user)
  LoginFragment -->>- UserServiceObserver :
  UserServiceObserver ->>+ LoginFragment : cancelMessage(user)
  LoginFragment -->>- UserServiceObserver :
  UserServiceObserver ->>+ LoginFragment : displayMessage("Hello " +
  user.getName())
  LoginFragment -->>- UserServiceObserver :
  deactivate UserServiceObserver
```

Questions

> Pick one place where you used the observer pattern. Which class was the subject?

`UserService`

> Which class was the observer?

`NavigateToUserObserver`

> Which layer did the subject belong to... (Model, View, or Presenter layer)

The "Model" layer

> ...and which layer did the observer belong to? (Model, View, or Presenter layer)

The "Presenter" layer

> Pick one place where you used generics. What class was it in?

`ResultHandler<ResultType, ObserverType extends ... >`

> What classes can the generic type T be?

`ResultType` is whatever the `ResultHandler` returns; the `ObserverType` is a `ResultObserver<ResultType>`

> Pick one place where you used the template method pattern. Show the template method.

```
protected void executeTask(Runnable task) {
    ExecutorService executor = Executors.newSingleThreadExecutor();
    executor.execute(task);
}
```

> What class is it in?

`BaseService`

> Show the step of the algorithm that is deferred to the class's subclass.

```
public void loginUser(
    String userAlias,
    String password,
    ResultObserver<Pair<User, AuthToken>> observer
) {
    LoginTask loginTask = new LoginTask(
        userAlias,
        password,
        new UserAuthHandler(observer)
    );
    executeTask(loginTask);
}
```

> What class is it in?

`UserService`