Refactorings

Clase Question Retriever:

Badsmell: Codigo duplicado.
Donde: En el metodo retrieveQuestions:, cuando se queda con las preguntas del dia de hoy. Esta linea aparece multiples a lo largo del metodo.
Refactoring: Extract Method. Se crea un metodo en la clase llamado todayQuestions, en todas las instancias donde se repetia ahora se llama al nuevo metodo.

Badsmell: Codigo duplicado.
Donde: En el metodo retrieveQuestions:, cuando ordena una coleccion de preguntas por la cantidad de votos. Esta linea aparece multiples a lo largo del metodo
Refactoring: extract method, se crea un metodo en la clase llamado sortQuestions, en todas las instancias donde se repetia ahora se llama al nuevo metodo.

Badsmell: Codigo duplicado.
Donde: En el metodo retrieveQuestions:, cuando se queda con los ultimos 100 elementos de la coleccion. Esta linea aparece multiples a lo largo del metodo
Refactoring: extract method, se crea un metodo en la clase llamado lastQuestions, en todas las instancias donde se repetia ahora se llama al nuevo metodo.

Badsmell: Codigo duplicado.
Donde: En el metodo retrieveQuestions:, cuando se queda solo con los elementos de la coleccion cuyo usuario no es el recibido como parametro. Esta linea aparece multiples a lo largo del metodo
Refactoring: extract method, se crea un metodo en la clase llamado rejectQuestions:FromUser:, en todas las instancias donde se repetia ahora se llama al nuevo metodo.

Badsmell: Reinventando la rueda.
Donde: En el metodo todayQuestions, usa do: junto con ifTrue: addAll: , cuando tendria que utilizar select:.
Refactoring: se reemplaza el bloque por un select:

Badsmell: Envidia de atributo.
Donde: En el metodo todayQuestions, cuando a cada pregunta envia el mensaje timestamp y a lo recibido le envia el mensaje asDate.
Refactoring: Move Method, se crea el metodo creationDate en Question para eliminar el: question timestamp asDate. Se reemplaza lo original por question creationDate.

Badsmell: Envidia de atributo.
Donde: en el metodo todayQuestions. El metodo utiliza su variable cuoora para obtener todas las preguntas realizadas y luego obtiene la fecha de creacion de cada una y se queda con la pregunta si fue creada hoy.
El metodo no tiene por que recibir toda esa informacion de cuoora, solo recibir las preguntas que se crearon hoy.
Refactoring: Move Method. Se mueve la implementacion a un nuevo metodo getTodayQuestions en cuoora. En el metodo original ahora se llama al creado en cuoora.

Badsmell: Envidia de atributo.
Donde: En el metodo sortQuestions. cuando ordena las preguntas por la cantidad de votos positivos, mediante un llamada a positiveVotes de cada pregunta y a lo devuelto le envia el mensaje size.
Refactoring: Move Method. Se crea un metodo en question, moviendo la linea de codigo a ese metodo, llamado totalPositiveVotes. En sortQuestions ahora se llama a ese metodo.

Badsmell: Switch
Donde: en el metodo retrieveQuestion:, tenemos 4 casos posibles, con implementaciones para cada caso, dependiendo del option del QuestionRetriever.
Refactoring: Replace conditional with polymorfism. Se crean 4 subclases de QuestionRetriever, una para cada opcion (SocialRetriever, TopicsRetriever, PopularTodayRetriever y NewsRetriever).
Los metodos que creamos para el codigo duplicado quedan en la clase QuestionRetriever y creamos metodos que redefinan al de la superclase.
Como en todas las opciones, una vez que conseguia las preguntas que queria hacia siempre lo mismo (sortQuestions: y luego lastQuestions: y por ultimo rejectQuestions:FromUser:), se decidio que el metodo a redefinir seria el que obtiene las preguntas que cada opcion quiere, por lo tanto en la super clase quedo definido el metodo retrieveQuestions: como:

```
retrieveQuestions: aUser
  | qRet |
  qRet := self getQuestionsFrom: aUser.
  qRet := self sortQuestions: qRet.
  qRet := self lastQuestions: qRet.
  ^self rejectQuestions: qRet FromUser: aUser.
```

Y siendo getQuestionsFrom: definido como:

```
getQuestionsFrom: aUser
  SubclassResponsability
```

Ahora seguimos el refactoring con Extract method, pero el codigo que queremos extraer tiene variables locales modificadas y por lo tanto primero tendremos que resolverlas antes de poder realizar el Extract method, mediante el refactoring Replace temp with query. Luego de resolverlas, extraemos el codigo resultante a getQuestionsFrom, de la siguiente manera:

- En caso de NewsRetriever (opcion #news), el query ya lo teniamos, todayQuestions, asi que se reemplaza por el llamada todayQuestions, se elimina newsCol y listo. Eso se extrae a getQuestionsFrom: aUser de NewsRetriever.
- Para PopularTodayRetriever (opcion #popularToday), se elimina la variable temporal averageVotes, creando el metodo averageVotes en la clase cuoora, y llamandolo en getQuestionsFrom. Para eliminar la variable popularCols se utiliza el metodo todayQuestions creado anteriormente.
- En caso de SocialRetriever (opcion #social), se elimina la envidia de atributo al mismo tiempo que eliminamos las variables temporales mediante la creacion del metodo questionsOfFollowing en la clase User, el cual devuelve las preguntas realizadas por los usuarios que sigue. En este metodo tambien se encontro el badsmell **Reinventando la rueda**, por lo que modificamos la implementacion para que use flatCollect. El metodo getQuestionsFrom de SocialRetriever llamara a ese metodo.
- Finalmente, en TopicsRetriever, se crea el metodo questionsOfTopic en la clase User (otra vez, para eliminar la envidia de atributo y eliminar la variable temporal) que devuelve las preguntas que contienen los topicos que le interesan al usuario. En este metodo tambien se encontro el badsmell **Reinventando la rueda**, por lo que modificamos la implementacion para que use flatCollect. El getQuestionsFrom llamara a este metodo.

Clase Question/Answer:

Badsmell: Codigo duplicado.
Donde: La clase Answer y la clase Question comparten la gran mayoria de los metodos, e incluso las variables de instancia. **Refactoring:** Extract superclass, se crea la superclase Publication, se mueven los metodos en comun ahi, junto con las varibales de instancia comunes a ambas clases. Las clases originales ahora son sublclases de Publication. Si bien la clase Answer parece que esta practicamente vacia, es porque casi todos los metodos los comparte con Question.

Por ultimo se modifico el test de la claser Question Retriever, para que el setup cree instancias de cada una de las subclases en vez de 4 questionRetrievers generales. Si bien en la implementacion final no es necesaria, se deja la variable de instancia option, en caso de ser necesaria a futuro, ya que estamos considerando que el proyecto no es uno terminado.