

Pasos iniciales

Para levantar la solución, ejecutar lo siguientes pasos a través de línea de comandos:

1. Descargar código fuente:



```
git clone https://github.com/flakorules/game-02-solution.git
cd game-02-solution
```

2. Instalar dependencias.



```
npm install
```

3. Ejecutar tests unitarios.

Sin cobertura de código:



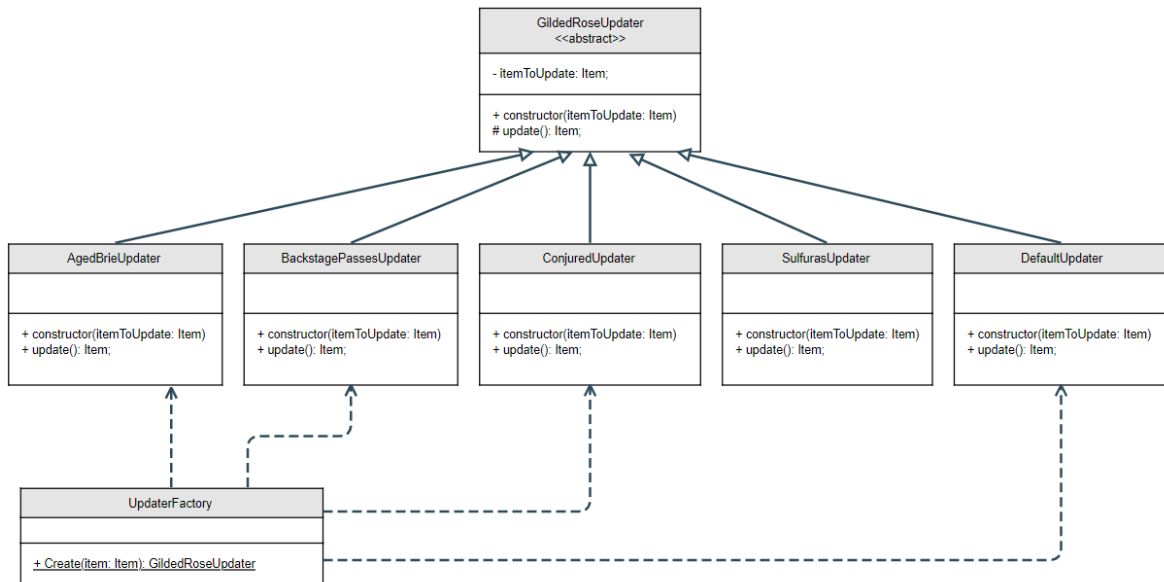
```
npm run test
```

Con cobertura de código:



```
npm run test-coverage
```

Diagrama de clases de la solución.

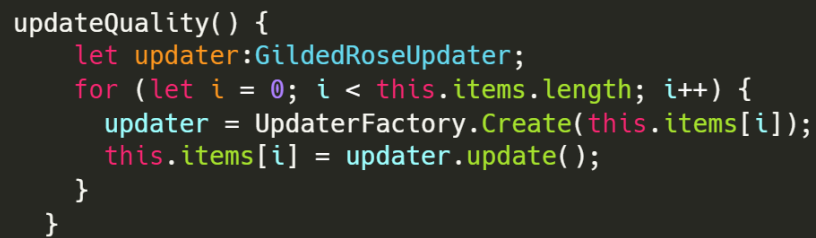


Descripción de la solución.

1. Dada la descripción del problema, se identifican 4 tipos de items de acuerdo a textos clave dentro de sus nombres:
 - > Aged Brie
 - > Backstage Passes
 - > Conjured
 - > Sulfuras
 - > Default
2. Una vez identificada esta clasificación, se crea la clase abstracta **GlidedRoseUpdater**, la cual declara como propiedad el item a actualizar (**itemToUpdate**) y el método abstracto **update**.
3. Se crean las clases **AgedBrieUpdater**, **BackstagePassesUpdater**, **ConjuredUpdater**, **SulfurasUpdater** y **DefaultUpdater** -que heredan de **GlidedRoseUpdater**-, las cuales implementan el método **update** con la lógica de actualización asociada a sus respectivos tipos de item.
4. Se crea la clase **UpdateFactory** con el método estático **Create** cuyo objetivo es instanciar y retornar el objeto de la clase concreta de acuerdo al tipo de item.

```
export class UpdaterFactory {  
  static Create(item: Item): GildedRoseUpdater {  
    if (item.name.indexOf("Aged Brie") >= 0)  
      return new AgedBrieUpdater(item);  
    else if (item.name.indexOf("Sulfuras") >= 0)  
      return new SulfurasUpdater(item);  
    else if (item.name.indexOf("Backstage passes") >= 0)  
      return new BackstagePassesUpdater(item);  
    else if (item.name.indexOf("Conjured") >= 0)  
      return new ConjuredUpdater(item);  
  
    return new DefaultUpdater(item);  
  }  
}
```

5. Se modifica **updateQuality** en clase **GildedRose**. Para cada elemento del array **items**:
- > Se invoca el método **Create** estático de la clase **UpdateFactory**, y se asigna el resultado al objeto **updater**.
 - > Se invoca el método **update** del objeto **updater**, y se sobre escribe el elemento del array **items**.



```
updateQuality() {  
  let updater:GildedRoseUpdater;  
  for (let i = 0; i < this.items.length; i++) {  
    updater = UpdaterFactory.Create(this.items[i]);  
    this.items[i] = updater.update();  
  }  
}
```