Orientación a Objetos II 2024

Explicación de práctica
Semana del 14 de abril

Ejercicios de la semana

- Ejercicio 4: ToDoItem
- Ejercicio 5: Decodificador de películas
- Ejercicio 6: Excursiones
- Ejercicio 7: Calculadora
- Ejercicio 8: Dispositivo móvil y conexiones
- Ejercicio 9: Alquiler de automóviles

Ejercicio 4 - ToDoItem

Este sistema permite definir tareas con nombre y una serie de comentarios.

Las tareas atraviesan diferentes etapas a lo largo de su ciclo de vida y ellas son:

- *pending*,
- *in-progress*,
- *paused* y
- *finished*.

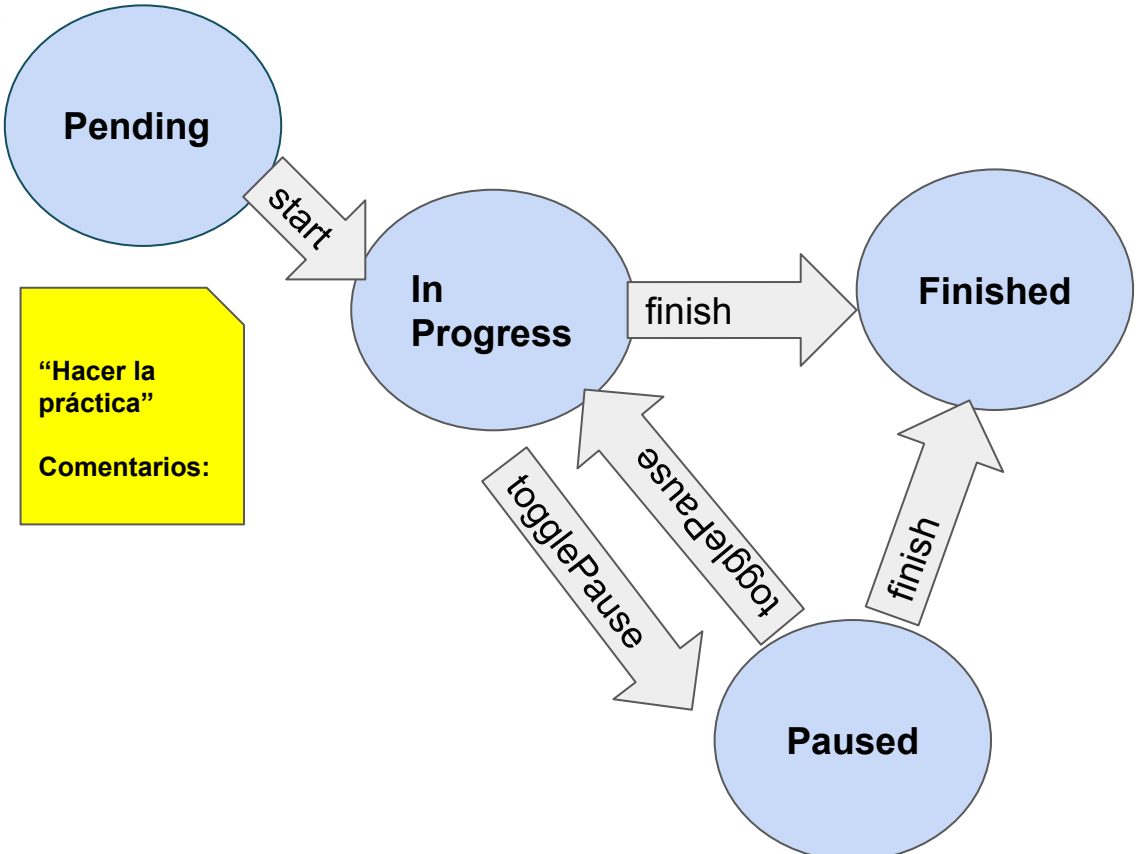
“Hacer la práctica”

Comentarios:
- Consultar Ej. 6
- Iniciar Ej. 7.

Ejercicio 4 - ToDoItem

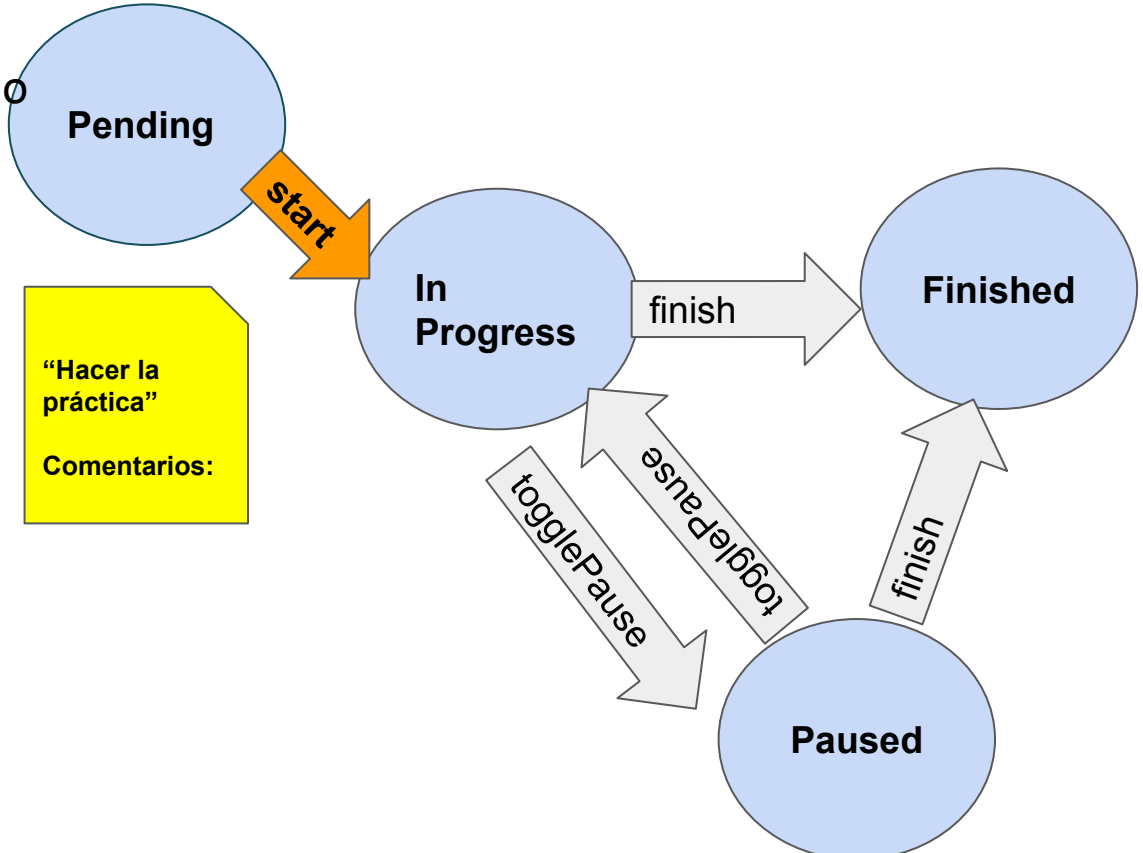
Cada tarea debe estar modelada mediante la clase **ToDoItem** con el siguiente protocolo:

```
//Instancia un ToDoItem
nuevo en estado pending con
<name> como nombre.

public ToDoItem(String name)
```



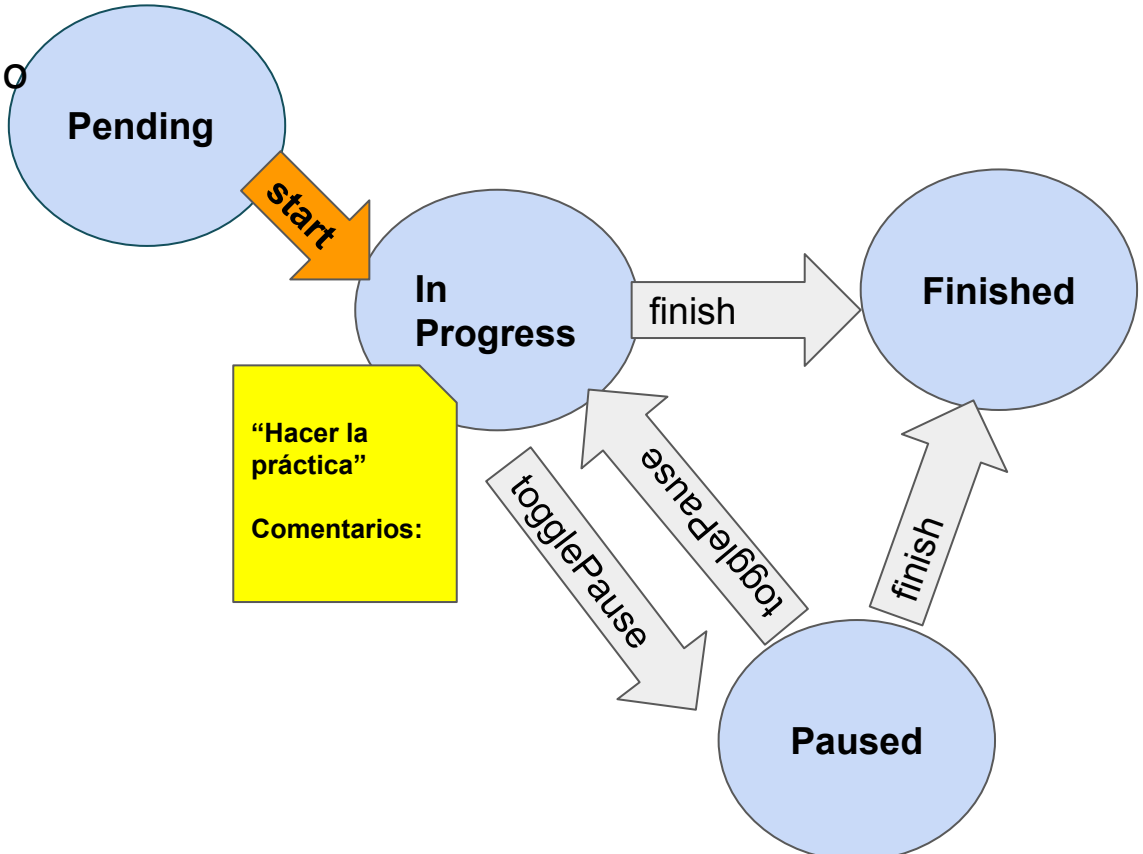
Ejercicio 4 - ToDoItem

```
// Pasa el ToDoItem a
in-progress (siempre y cuando
su estado actual sea
pending, si se encuentra en
otro estado, no hace nada)

public void start()
```



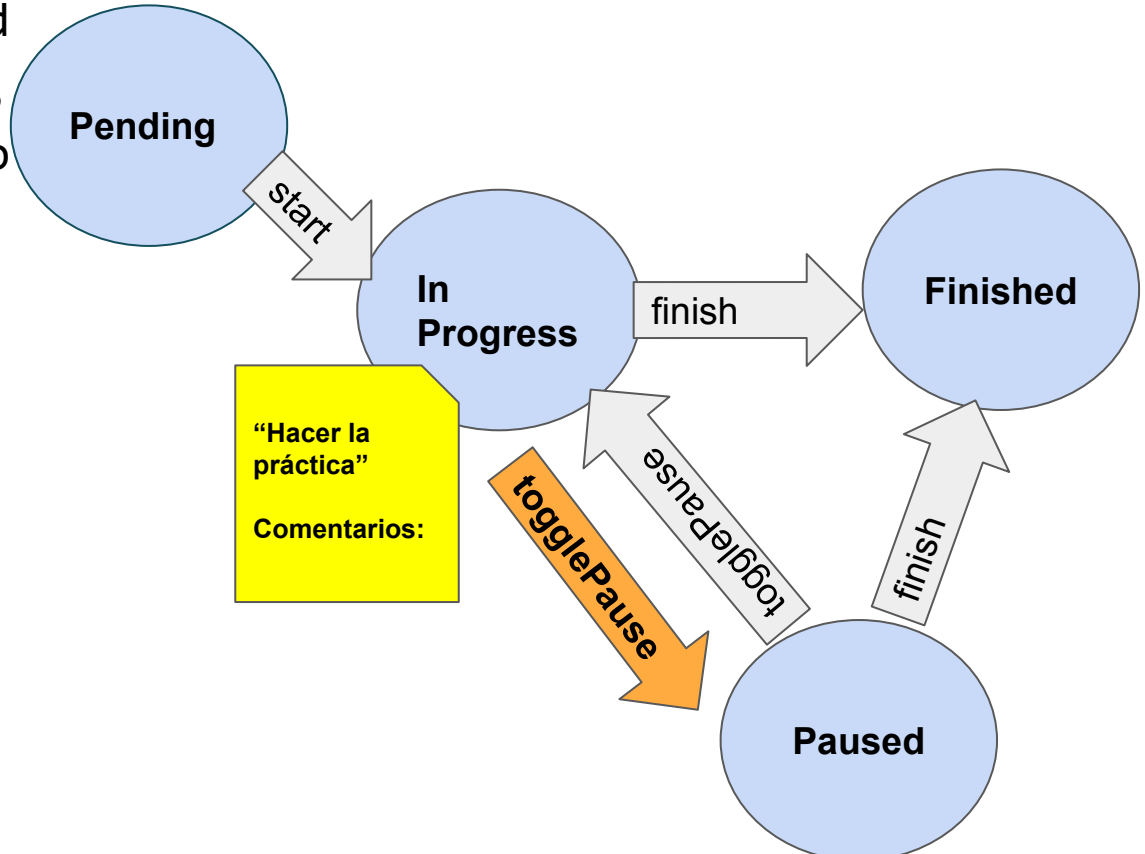
Ejercicio 4 - ToDoItem

```
// Pasa el ToDoItem a
in-progress (siempre y cuando
su estado actual sea
pending, si se encuentra en
otro estado, no hace nada)

public void start()
```



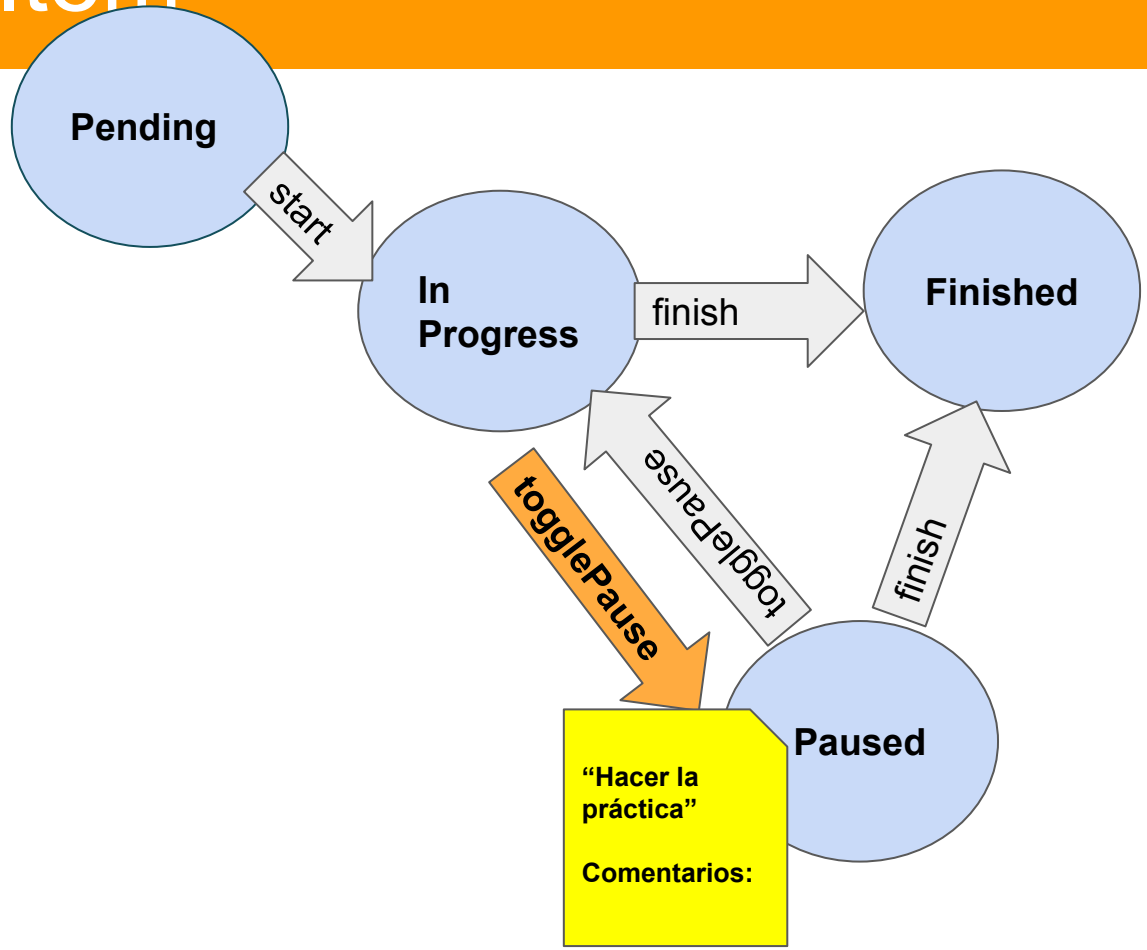
Ejercicio 4 - ToDoItem

```
// Pasa el ToDoItem a paused
si su estado es in-progress,
o a in-progress si su estado
es paused.
Caso contrario (pending o
finished) genera un error
informando la causa
específica del mismo

public void togglePause()
```



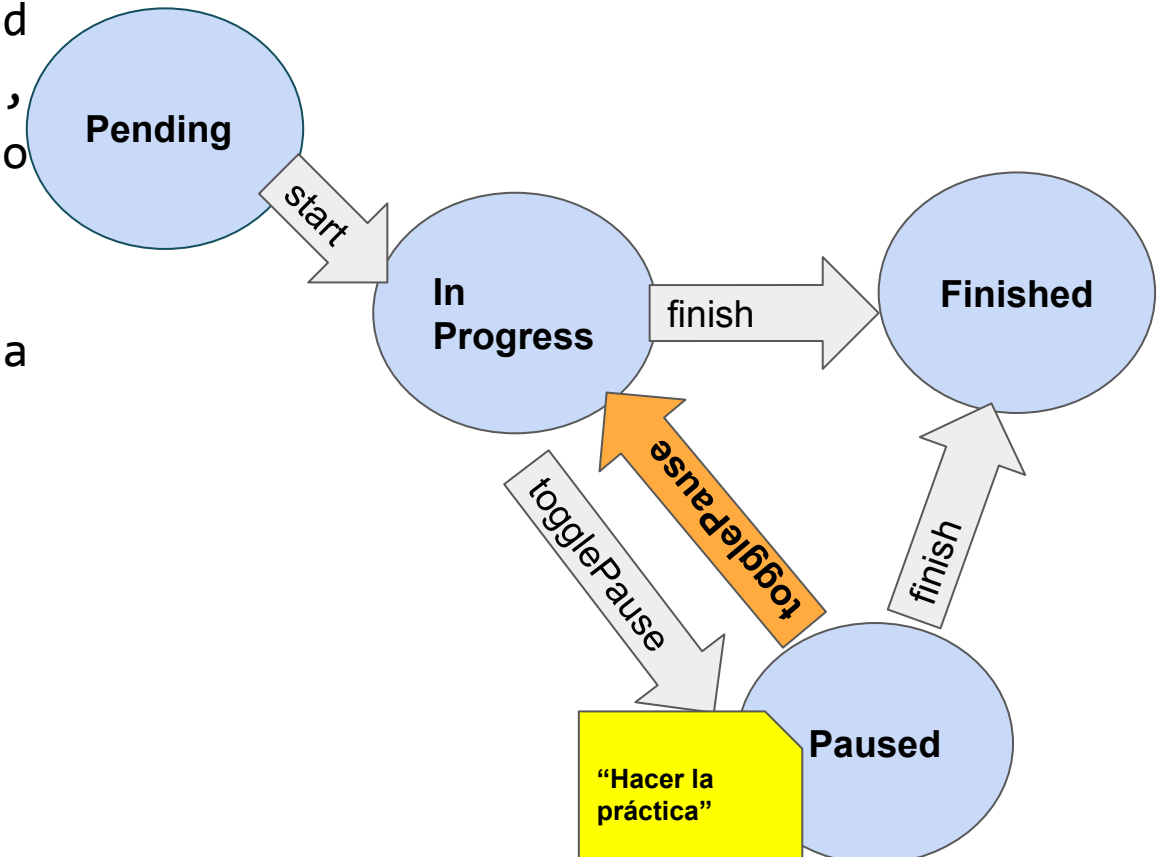
Ejercicio 4 - ToDoItem

```
// Pasa el ToDoItem a paused
si su estado es in-progress,
o a in-progress si su estado
es paused. Caso contrario
(pending o finished) genera
un error informando la causa
específica del mismo

public void togglePause()
```



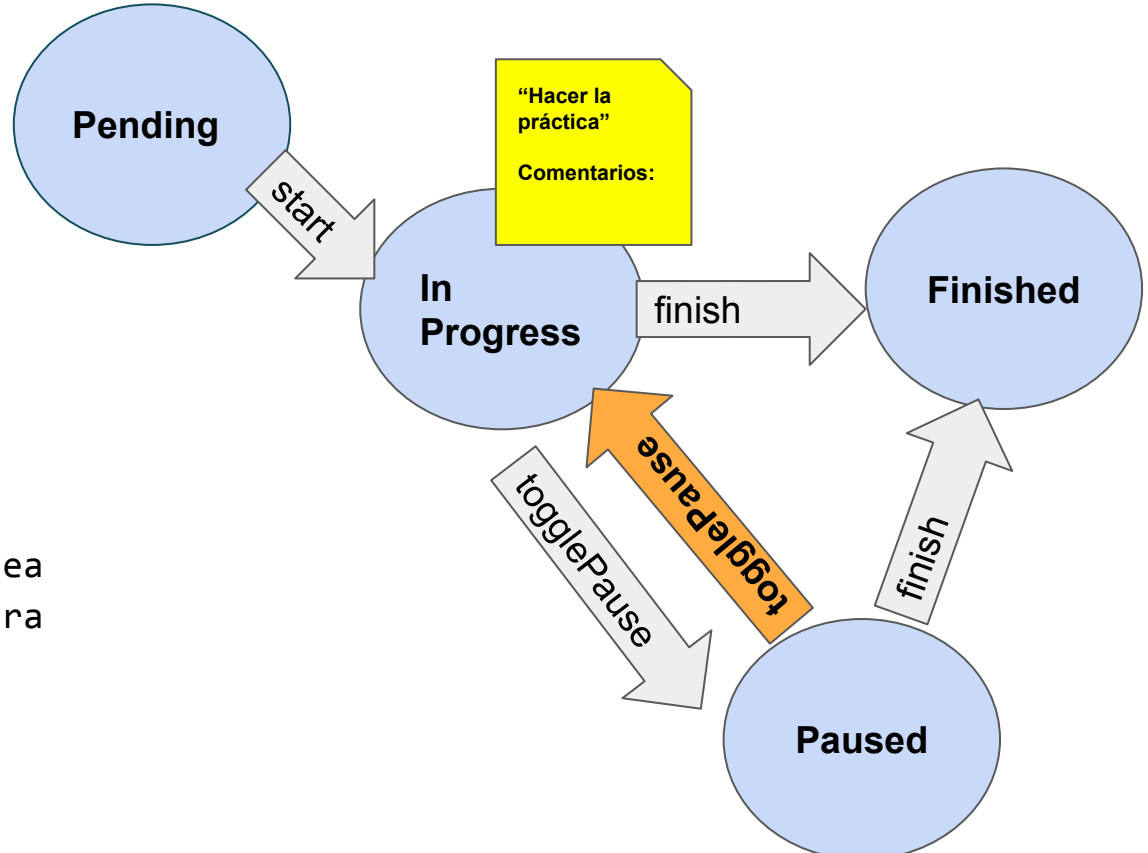
Ejercicio 4 - ToDoItem

```
// Pasa el ToDoItem a paused
si su estado es in-progress,
o a in-progress si su estado
es paused. Caso contrario
(pending o finished) genera
un error informando la causa
específica del mismo

public void togglePause()
```



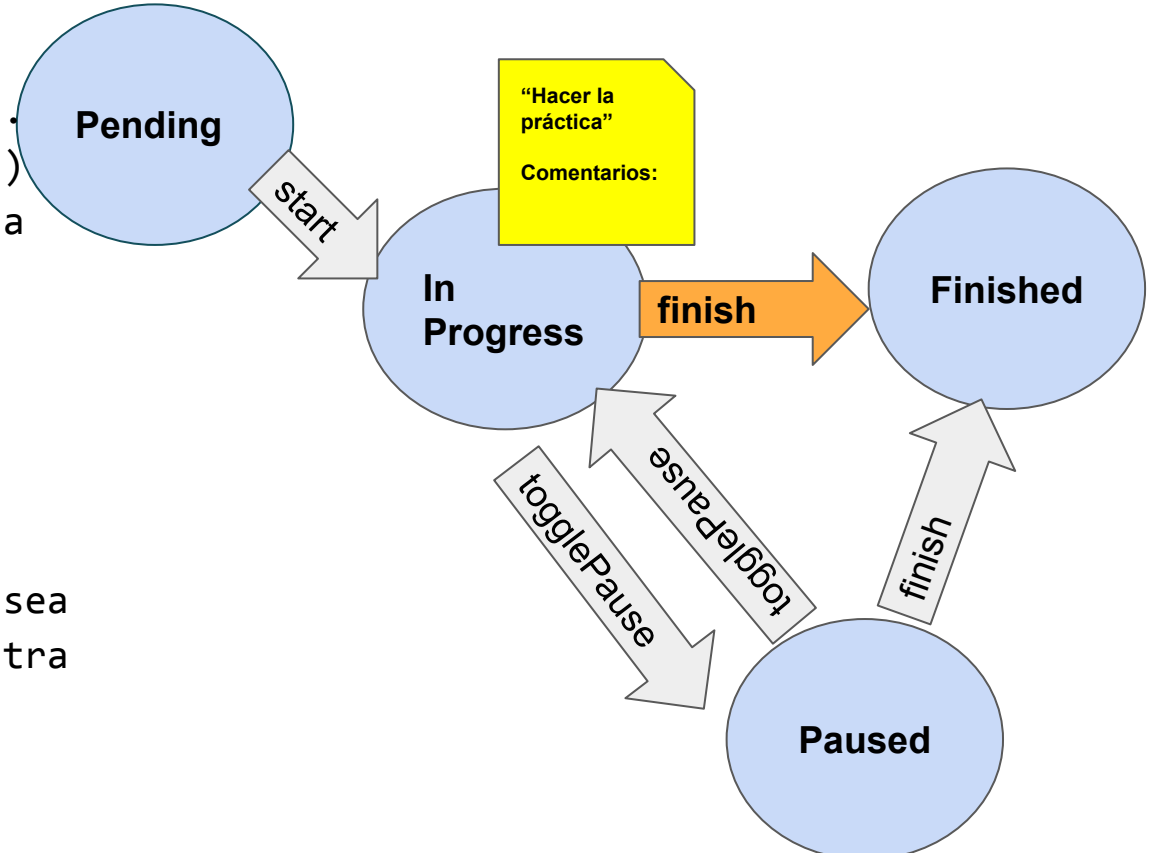
Ejercicio 4 - ToDoItem

```
// Pasa el ToDoItem a paused si su
estado es in-progress, o a
in-progress si su estado es paused.
Caso contrario (pending o finished)
genera un error informando la causa
específica del mismo

public void togglePause()


//Pasa el ToDoItem a finished,
siempre y cuando su estado actual sea
in-progress o paused. Si se encuentra
en otro estado, no hace nada.

public void finish()
```



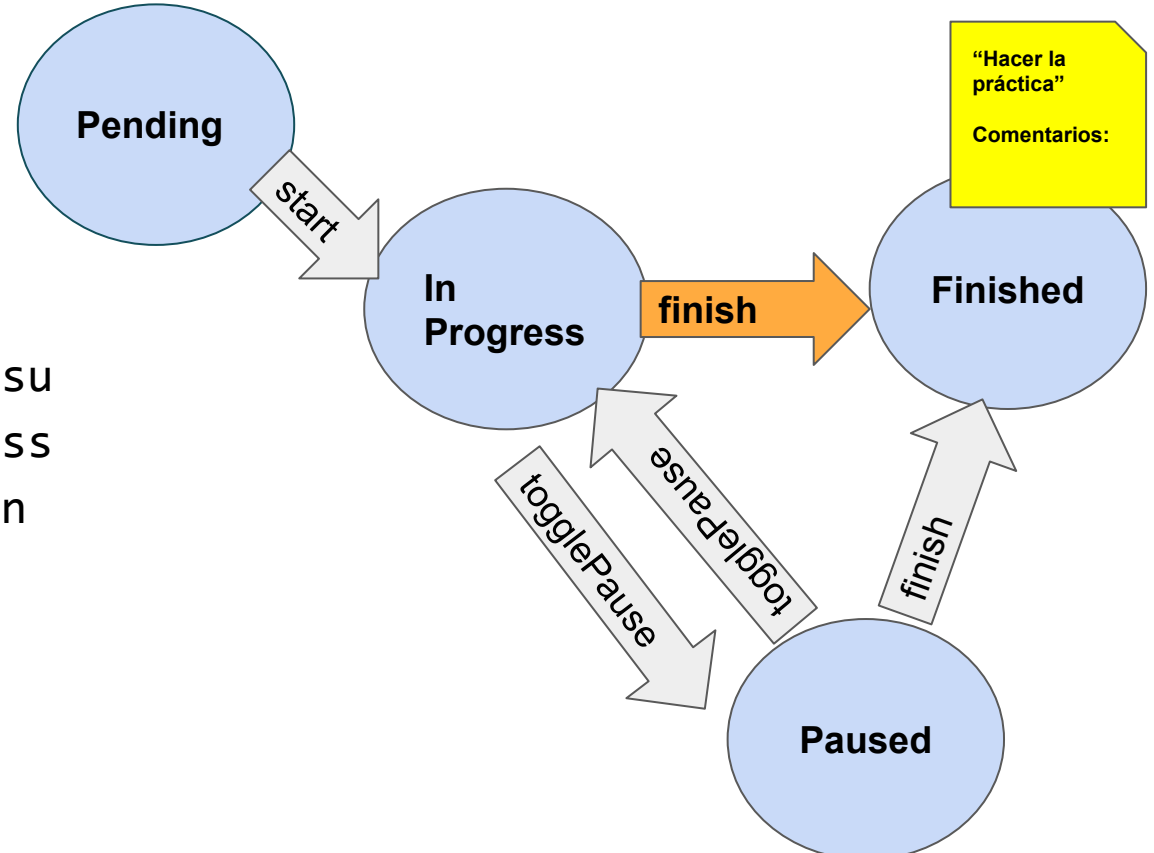
Ejercicio 4 - ToDoItem

```
// Pasa el ToDoItem a paused si su
estado es in-progress, o a
in-progress si su estado es paused.
Caso contrario (pending o finished)
genera un error informando la causa
específica del mismo

public void togglePause()


//Pasa el ToDoItem a finished,
siempre y cuando su estado actual sea
in-progress o paused. Si se encuentra
en otro estado, no hace nada.

public void finish()
```



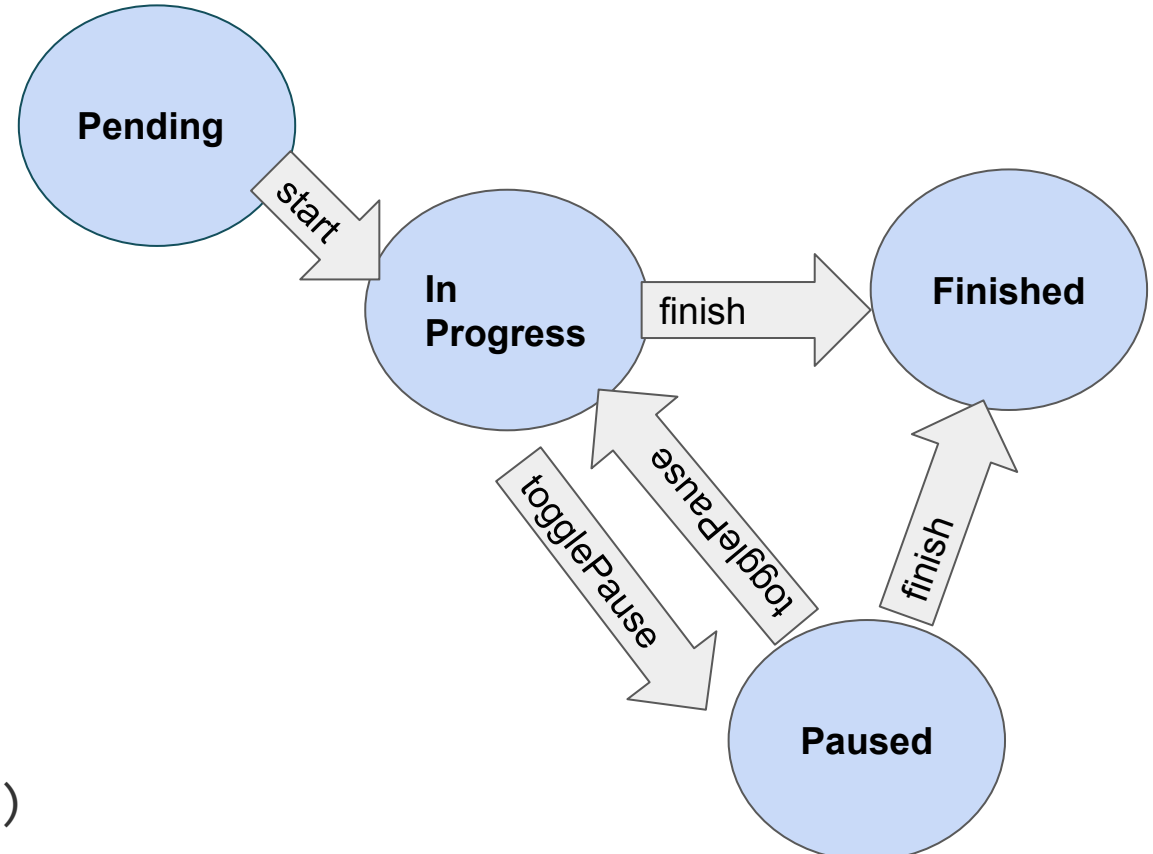
Ejercicio 4 - ToDoItem

```
//Pasa el ToDoItem a
finished, siempre y cuando su
estado actual sea in-progress
o paused. Si se encuentra en
otro estado, no hace nada.

public void finish()
```



Ejercicio 4 - ToDoItem

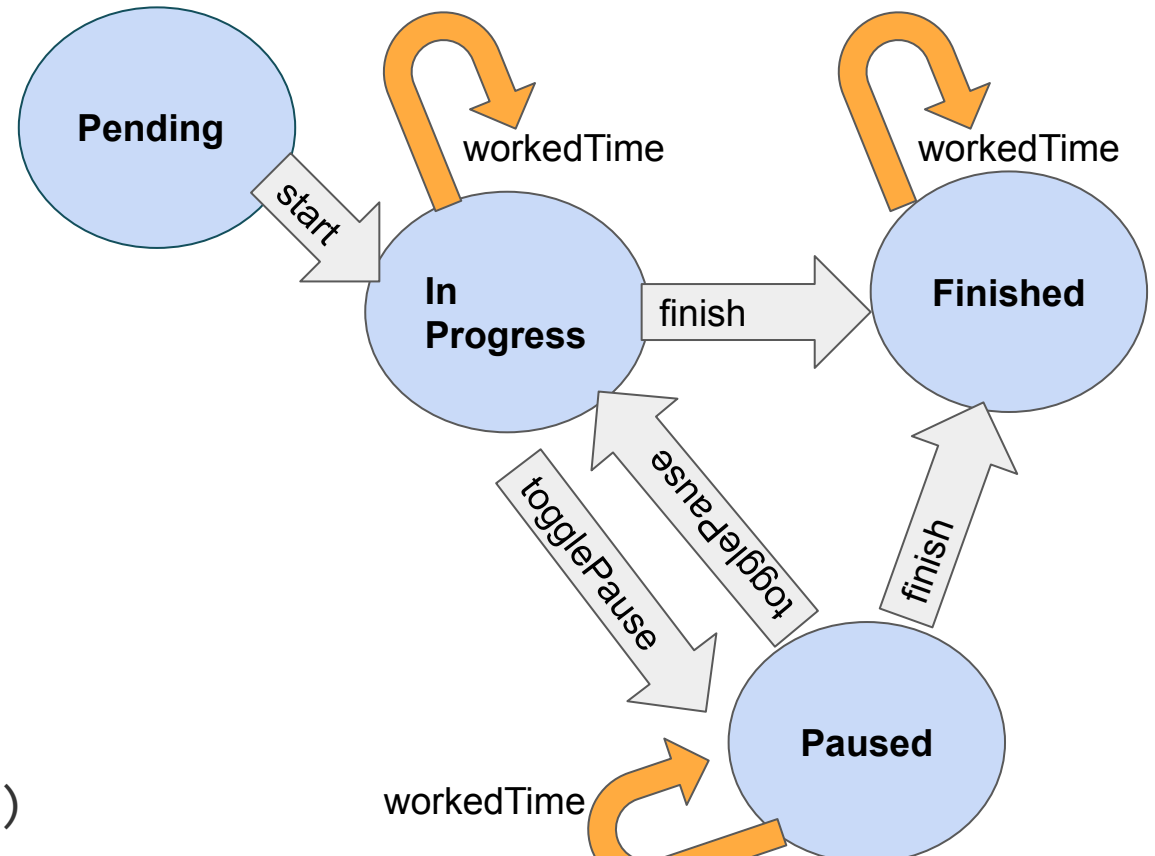
```
//Retorna el tiempo que
transcurrió desde que se
inició el ToDoItem (start)
hasta que se finalizó. En
caso de que no esté
finalizado, el tiempo que
haya transcurrido hasta el
momento actual. Si la tarea
no se inició, genera un
error informando la causa
específica del mismo.

public Duration workedTime()
```



Ejercicio 4 - ToDoItem

```
//Retorna el tiempo que
transcurrió desde que se
inició el ToDoItem (start)
hasta que se finalizó. En
caso de que no esté
finalizado, el tiempo que
haya transcurrido hasta el
momento actual. Si la tarea
no se inició, genera un
error informando la causa
específica del mismo.
```

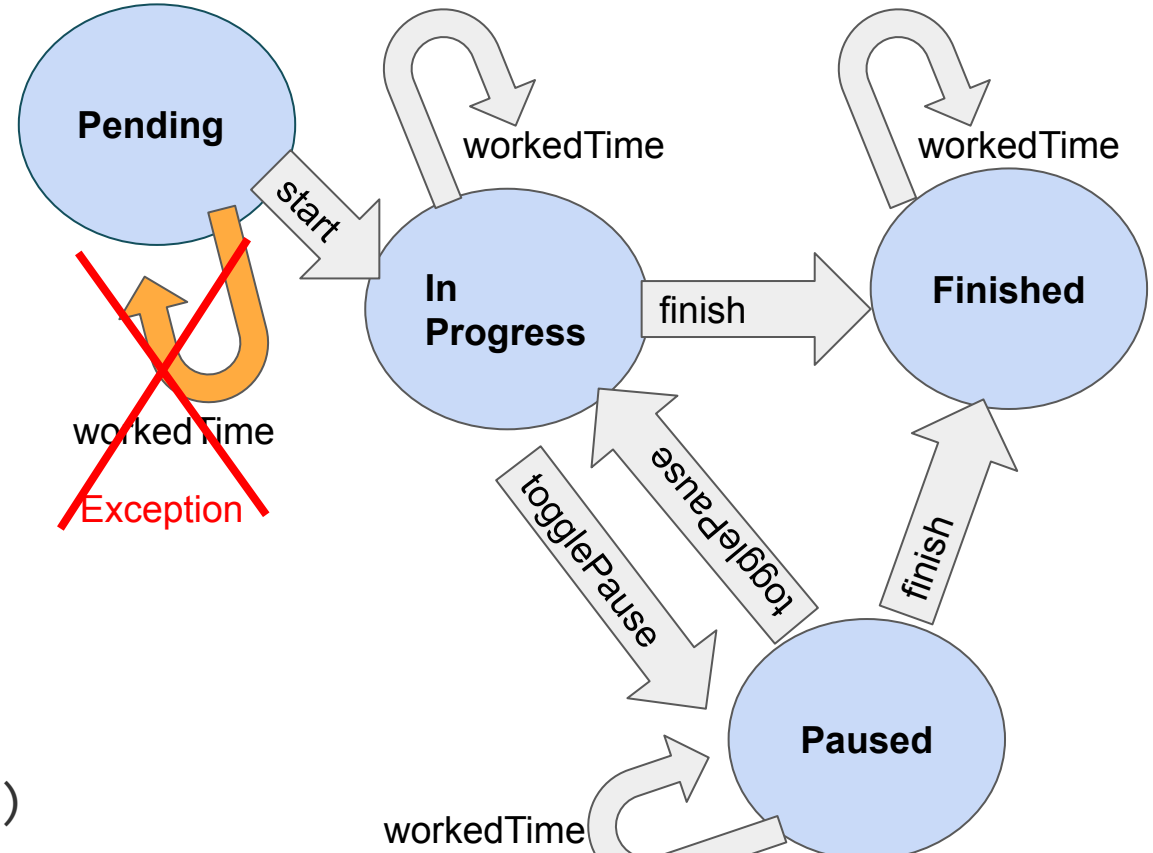
```
public Duration workedTime()
```



Ejercicio 4 - ToDoItem

```
//Retorna el tiempo que
transcurrió desde que se
inició el ToDoItem (start)
hasta que se finalizó. En
caso de que no esté
finalizado, el tiempo que
haya transcurrido hasta el
momento actual. Si la tarea
no se inició, genera un
error informando la causa
específica del mismo.
```

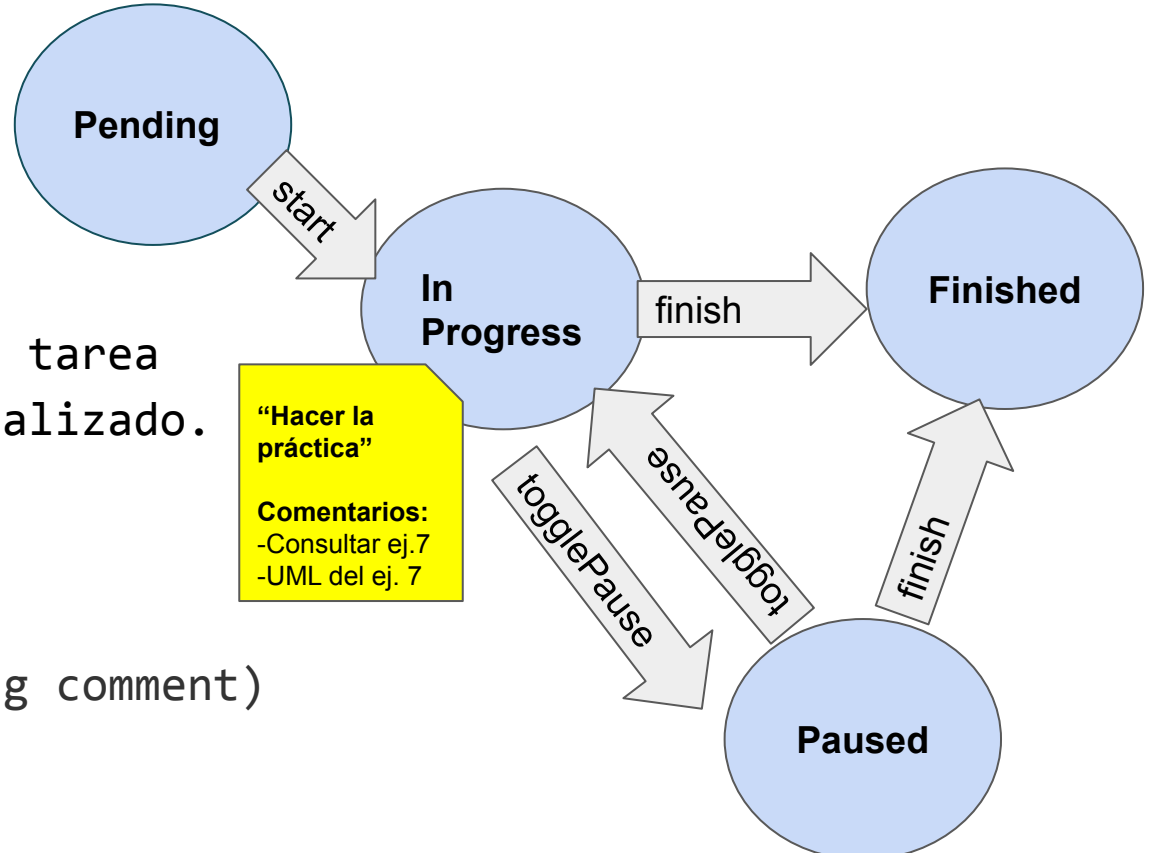
```
public Duration workedTime()
```



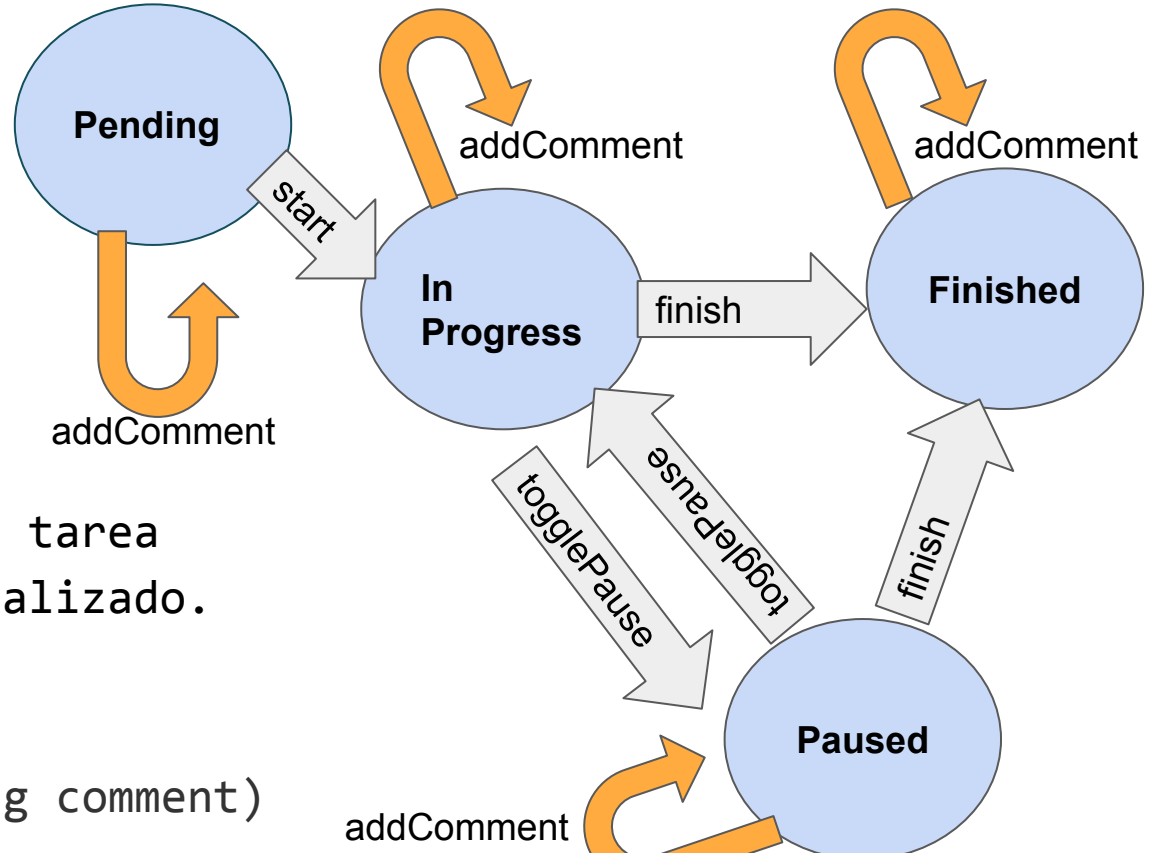
Ejercicio 4 - ToDoItem

```
// Agrega un comentario a la tarea
siempre y cuando no haya finalizado.
Caso contrario no hace nada.
```

```
public void addComment(String comment)
```



Ejercicio 4 - ToDoItem



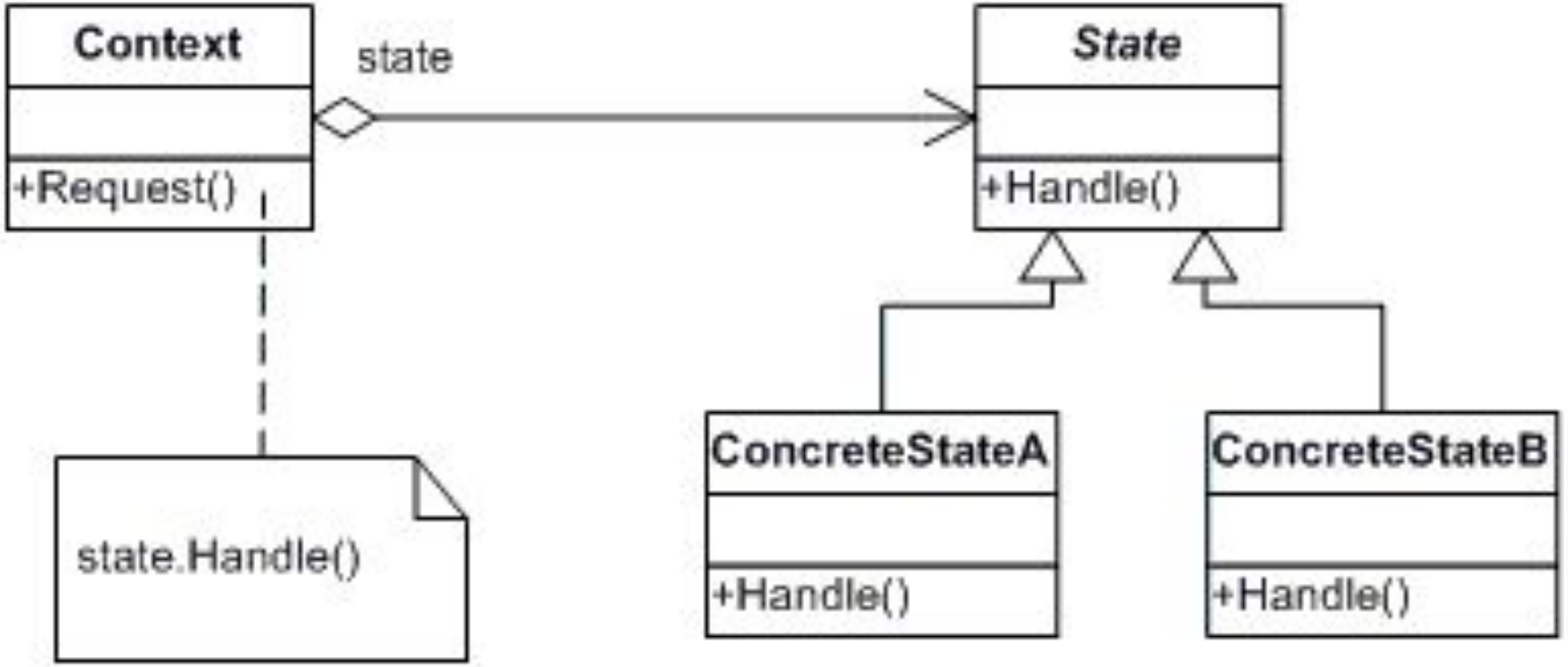

```
// Agrega un comentario a la tarea
siempre y cuando no haya finalizado.
Caso contrario no hace nada.

public void addComment(String comment)
```

Ejercicio 4 - ToDoItem

¿Conocen algún patrón de diseño que se pueda utilizar en esta situación?

Ejercicio 4 - ToDoItem

Diagrama de Patrón de diseño State:



Ejercicio 4 - ToDoItem

```
public class ToDoItem {
  private String name;
  private ToDoItemState state;
  . . .

  public void togglePause() {
    state.togglePause();
  }

  . . .
}
```

Ejercicio 4 - ToDoItem

```
class ToDoItem {
 private String name;
 private ToDoItemState state;
 . . .

 public void togglePause() {
   state.togglePause();
 }

 . . .
}
```

```
class InProgress extends ToDoItemState
{
   . . .

   public void togglePause() {
     ...
   }

   . . .
 }
```

¿Cómo se logra que el ToDoItem (su contexto) pase al estado Paused?

Ejercicio 4 - ToDoItem

```
class ToDoItem {
 private String name;
 private ToDoItemState state;
 . . .

 public void togglePause() {
   state.togglePause(this);
 }

 . . .
}
```

```
class InProgress extends ToDoItemState
{
  . . .

  public void togglePause(
    ToDoItem context) {
     context.setStatus(new Paused());
   }

  . . .

}
```

Ejercicio 4 - ToDoItem

¿Hay alguna otra forma de que se conozcan contexto-estado?

Ejercicio 5 - Decodificador de Películas

Sistema de alquileres de películas *on-demand* que ofrece sugerencias a partir de lo visto por el usuario.

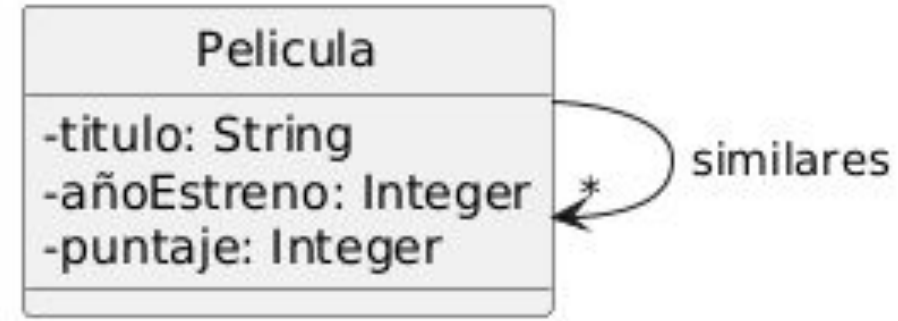
El “decodificador” conoce la lista completa de películas, así como las que vio el usuario.

Tiene un mecanismo para sugerir contenido.

Ejercicio 5 - Decodificador de Películas

De cada película conocemos:

- Título
- Año de su estreno
- Puntaje
- Películas similares
  Esta es una relación recíproca.



Ejercicio 5 - Decodificador de Películas

El decodificador tiene los siguientes mecanismos de sugerencia para ofrecer 3 películas no reproducidas:

- **Novedad**: Ofrece películas ordenadas por su fecha de estreno.
- **Similaridad**: Similares a las reproducidas, ordenadas por su fecha de estreno.
- **Puntaje**: Ofrece películas con mayor puntuación. Y en caso de empate, se considera la más reciente.

Este criterio no es fijo y se puede cambiar en cualquier momento!

Ejercicio 5 - Decodificador de Películas

Diagrama de Patrón de diseño Strategy: