Interaction Diagrams (Interact with Object)

Group 6

Blekinge Institute of Technology

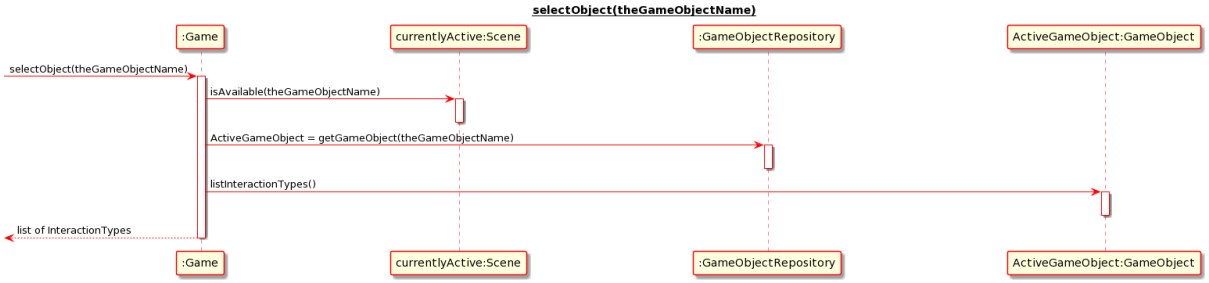
PA1472

10/02/2023

Oliver Bölin, Farhad Asadi, Samuel Täng, Michael Törnvall, Kim Budischewski

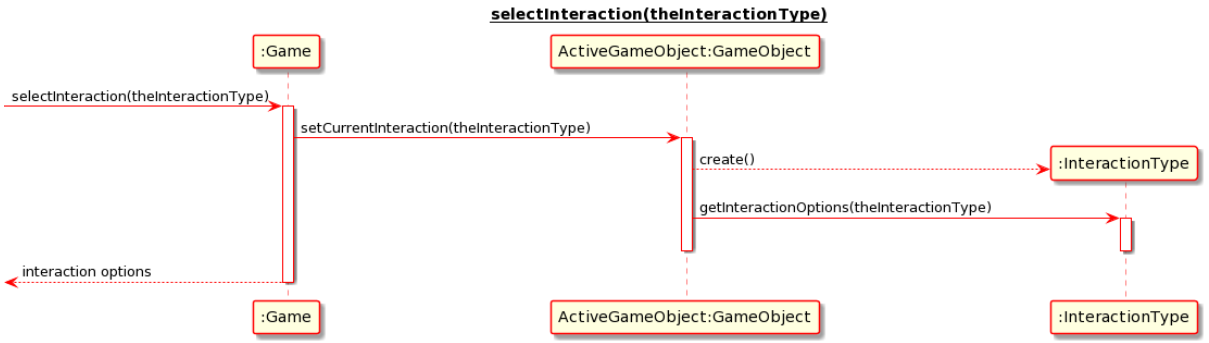
SelectObject(theGameObjectName)

In this use case the *:Game* communicates with the controller of this use case. The game is using *currentlyActive:Scene* as an information expert to get information if the object/s is available in the scene.
Then we pass it to *:GameRepository* that is an information expert on all the objects in total. Now *:Game* can communicate with *ActiveGameObject:GameObject* to get a list of interaction types, and therefore return the list.



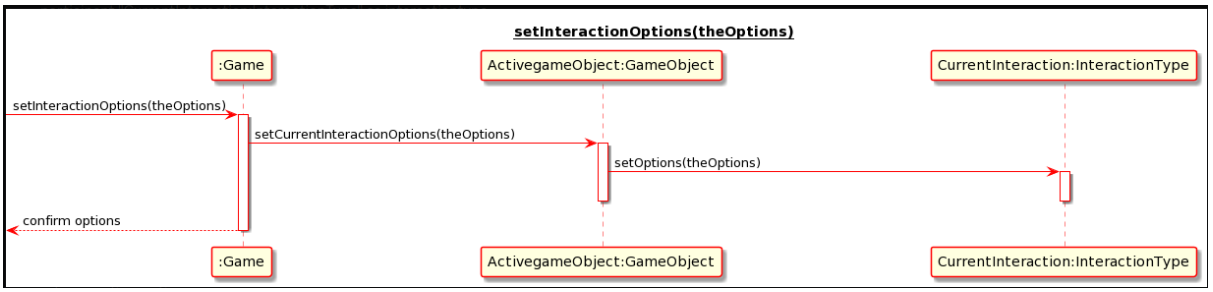
SelectInteractions(theInteractionType)

:Game communicates with *ActiveGameObject:GameObject* which is the creator of *InteractionType*.
The created *InteractionType* is the information expert on what options are available for it, so we ask it for the options it has right now. The available options get returned.



setInteractionOptions(TheOptions)

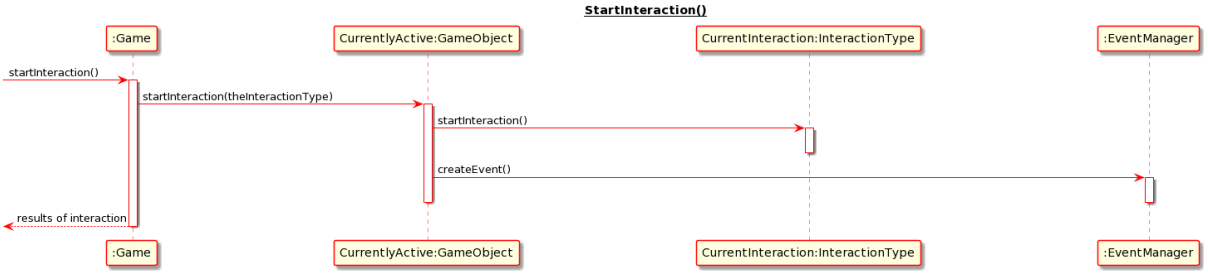
In this use case the :*Game* passes the *theOptions* to the *GameObject* object thus the *Game* class's primary responsibility becomes the management of the game state. The class *ActivegameObject* holds the information about the current active game object and it is the information expert to set the interaction options for the active game object. Hence, the *ActivegameObject* class is responsible for setting the current interaction options for the active game object.



startInteraction()

The startInteraction function is used to start the interaction with the selected interaction options. Then createEvent() sends a request to the EventManager who creates the event startInteraction() and returns the results of the interaction.

Event is the controller of startInteraction and it receives and coordinates the controls for when to start an Interaction. If you assign the task of creating new events to gameElement, you get low coupling, but giving it to Events gives us even lower coupling.



UML-diagrams code

selectObject(theGameObjectName)

```
@startuml
title <u>selectObject(theGameObjectName)</u>
end title
skinparam participantpadding 100
skinparam ParticipantBackgroundColor LightYellow
skinparam ParticipantBorderColor red
skinparam ParticipantBorderThickness 1.5
skinparam Shadowing true
skinparam ArrowColor red
skinparam SequenceLifeLineBorderColor red
participant ":Game" as game
participant "currentlyActive:Scene" as scene
participant ":GameObjectRepository" as objectRepository
participant "ActiveGameObject:GameObject" as object
->game:selectObject(theGameObjectName)
activate game
game->scene:isAvailable(theGameObjectName)
activate scene
deactivate scene
game->objectRepository:ActiveGameObject =
getGameObject(theGameObjectName)
activate objectRepository
deactivate objectRepository
game->object:listInteractionTypes()
activate object
deactivate object
return list of InteractionTypes
deactivate game
@enduml
```

selectInteraction(theInteractionType)

```
@startuml
title
 <u>selectInteraction(theInteractionType)</u>
end title
```

```
skinparam Participantpadding 100
skinparam ParticipantBackgroundColor LightYellow
skinparam ParticipantBorderColor red
skinparam ParticipantBorderThickness 1.5
skinparam Shadowing true
skinparam ArrowColor red
skinparam SequenceLifeLineBorderColor red
scale 1
participant ":Game" as game
participant "ActiveGameObject:GameObject" as object
participant ":InteractionType" as interactionType
->game:selectInteraction(theInteractionType)
activate game
game->object:setCurrentInteraction(theInteractionType)
activate object
object-->interactionType**:create()
"object"->interactionType:getInteractionOptions(theInteractionType)
activate interactionType
deactivate interactionType
deactivate object
return interaction options
@enduml
```

setInteractionOptions(theOptions)

```
@startuml
title
 <u>setInteractionOptions(theOptions)</u>
end title
skinparam Participantpadding 100
skinparam ParticipantBackgroundColor LightYellow
skinparam ParticipantBorderColor red
skinparam ParticipantBorderThickness 1.5
skinparam Shadowing true
skinparam ArrowColor red
skinparam SequenceLifeLineBorderColor red
scale 1
participant ":Game" as game
participant "ActivegameObject:GameObject" as object
participant "CurrentInteraction:InteractionType" as interactiontype
->game:setInteractionOptions(theOptions)
activate game
game->object:setCurrentInteractionOptions(theOptions)
activate object
object->interactiontype:setOptions(theOptions)
```

```
activate interactiontype
deactivate interactiontype
deactivate object
return confirm options
@enduml
```

startInteraction()

```
@startuml
title
 <u>StartInteraction()</u>
end title
skinparam Participantpadding 100
skinparam ParticipantBackgroundColor LightYellow
skinparam ParticipantBorderColor red
skinparam ParticipantBorderThickness 1.5
skinparam Shadowing true
skinparam ArrowColor red
skinparam SequenceLifeLineBorderColor red
scale 1
participant ":Game" as game
participant "CurrentlyActive:GameObject" as GO
participant "CurrentInteraction:InteractionType" as IT
participant ":EventManager" as Event
->game:startInteraction()
activate game
game->GO:startInteraction(theInteractionType)
activate GO
GO->IT:startInteraction()
activate IT
deactivate IT
GO->Event:createEvent()
activate Event
deactivate Event
deactivate GO
return results of interaction
deactivate game
@enduml
```