

Description

This blank UML template can help you:

• Access shapes to create UML activity, sequence, state machine, or use case diagrams

• Describe the boundary, structure, and behavior of a system and its objects

• Create a UML diagram starting from a blank canvas

Add content to customize this blank canvas to your use case.

Tutorials

(Hold Shift + ⌘ or Ctrl, then click)

Watch Lucidchart basic tutorials

```
classDiagram
    class Inek {
        +InekSutMiktari{get;set;}:int
        +Inek(progressBar:ProgressBar)
    }
    class Keci {
        +KeciSutMiktari{get;set;}:int
        +Keci(progressBar:ProgressBar)
    }
    class Hayvan {
        <<abstract>>
        +ilkOlum:bool
        +ses:SoundPlayer
        +progressBar:ProgressBar
        +CanliMi{get;set;}:bool
        +yemVer():void
        +canAzalt():void
        +sesCal():void
    }
    class Ordek {
        +OrdekYumurtaSayisi{get;set;}:int
        +Ordek(progressBar:ProgressBar)
    }
    class Tavuk {
        +TavukYumurtaSayisi{get;set;}:int
        +Tavuk(progressBar:ProgressBar)
    }
    class IAhirHayvani {
        <<interface>>
        +SutSat():int
        +SutUret():int
    }
    class IKumesHayvani {
        <<interface>>
        +YumurtaSat():int
        +YumurtaUret():int
    }
    class Oyun {
        +IblSaniye:Label
        +IblTYAdet:Label
        +IblOYAdet:Label
        +IblISKG:Label
        +IblKSKG:Label
        +IblITL:Label
        +Oyun(progressBar prgbarTavuk, ProgressBar prgbarOrdek, ProgressBar prgbarInek, ProgressBar prgbarKeci, Label IblSaniye, Label IblTYAdet, Label IblOYAdet, Label IblISKG, Label IblKSKG, Label IblITL):
        +OyunuOynat():void
        +YemVerTavuk():void
        +TavukYumurtasiSat():void
        +YemVerOrdek():void
        +OYSat():void
        +YemVerInek():void
        +ISSat():void
        +KSSat():void
    }

    Inek --> Keci
    Keci --> Hayvan
    Hayvan <|-- Ordek
    Hayvan <|-- Tavuk
    Ordek ..> Hayvan
    Tavuk ..> Hayvan
    Ordek ..> IAhirHayvani
    Tavuk ..> IKumesHayvani
    Oyun --> Inek
    Oyun --> Keci
    Oyun --> Ordek
    Oyun --> Tavuk
```

The diagram illustrates a system architecture for a farm simulation. It features several classes and interfaces. The **Hayvan** class is an abstract base class for **Ordek** and **Tavuk**. **Ordek** and **Tavuk** implement methods from **IAhirHayvani** and **IKumesHayvani** respectively. The **Oyun** class is a central component that interacts with **Inek**, **Keci**, **Ordek**, and **Tavuk**. It manages game state and provides methods for playing the game and managing resources.