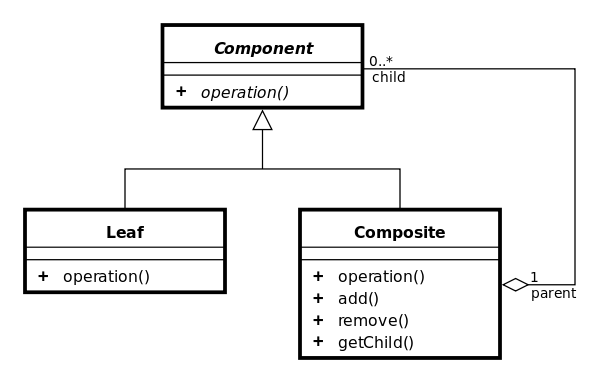
**Composite TasarımKalıbı**

Composite tasarım kalıbı bir grup nesnenin sanki tek bir nesneymiş gibi ele alınmasını sağlar.Sistemin bütününü oluşturan ağaç yapısındaki parçaların kendileri de alt parçalardan oluşabilir.Bu tasarım kalıbı sayesinde kullanıcı tek bir parça ile ilgileniyormuş gibi uygulama yazabilir.

**Ne ZamanKullanılır**

Kullanıcının sistemi oluşturan parçalar seviyesine inmeden bir bütün olarak sistemi kullanmayı istemesi durumunda bu kalıp kullanılır.

**Yapı**

[](http://en.wikipedia.org/wiki/File:Composite_UML_class_diagram_(fixed).svg)

[http://bits.wikimedia.org/static-1.22wmf1/skins/common/images/magnify-clip.png](http://en.wikipedia.org/wiki/File:Composite_UML_class_diagram_(fixed).svg)

|  |
| --- |
| **publicinterface** Parca {  **publicint** fiyat();  } |

|  |
| --- |
| **import** java.util.ArrayList;  **publicclass** Bilgisayar **implements** Parca {  **private** ArrayList<Parca>parcalar = **new** ArrayList<Parca>();  **publicvoid** ekle(Parca parca) {  parcalar.add(parca);  }  **publicvoid** sil(Parca parca) {  **if** (parcalar.contains(parca))  parcalar.remove(parca);  }  **public** Parca parcaVer(**int** no) {  Parca parca = **null**;  parca = parcalar.get(no);  **if** (parca == **null**)  **thrownew** ParcaYokHatasi(no + " nolu parca bulunamadi..");  **return** parca;  }  **public** ArrayList<Parca> parcaListesi() {  **return**parcalar;  }  @Override  **publicint** fiyat() {  **int**fiyat = 0;  **for** (**int** i = 0; i <parcalar.size(); i++) {  fiyat += parcalar.get(i).fiyat();  }  **return**fiyat;  }  } |

|  |
| --- |
| **publicclass**ParcaYokHatasi**extends**RuntimeException {  **public** ParcaYokHatasi(String mesaj) {  **super**(mesaj);  }  } |

|  |
| --- |
| **publicclass** Monitor **implements** Parca{  @Override  **publicint** fiyat() {  // **TODO** Auto-generated method stub  **return** 250;  }  } |

|  |
| --- |
| **publicclass** Fare **implements** Parca {  @Override  **publicint** fiyat() {  // **TODO** Auto-generated method stub  **return** 25;  }  } |

|  |
| --- |
| **publicclass** Klavye **implements** Parca {  @Override  **publicint** fiyat() {  // **TODO** Auto-generated method stub  **return** 50;  }  } |

|  |
| --- |
| **publicclass** Ram **implements** Parca {  @Override  **publicint** fiyat() {  // **TODO** Auto-generated method stub  **return** 100;  }  } |

|  |
| --- |
| **publicclass** Program {  /\*\*  \* **@param** args  \*/  **publicstaticvoid** main(String[] args) {  Bilgisayar bilgisayar = **new** Bilgisayar();  Parca monitor = **new** Monitor();  Parca klavye = **new** Klavye();  Parca fare=**new** Fare();  bilgisayar.ekle(monitor);  bilgisayar.ekle(fare);  bilgisayar.ekle(klavye);  System.*out*.println("monitor fiyati:"+monitor.fiyat());  System.*out*.println("bilgisayar fiyati:"+bilgisayar.fiyat());  }  } |

**Örnek**

Aşağıdaki örnek grafik sıfına bir örnektir. Belirli bir grafik daire, dikdörtgen vb alt grafiklerin yardımıyla oluşturulur.Asagidaki örnekte sadece Ellipse sınıfı ve print() metodu bulunmaktadır. translate(), move() vb metodlar eklenebilir.

|  |
| --- |
| /\*\* "Component" \*/  **Public interface** Graphic {  // Prints the graphic.    **public void** print();  } |

|  |
| --- |
| /\*\* "Composite" \*/  **import**java.util.List;  **import**java.util.ArrayList;  **class**CompositeGraphic**implements** Graphic {  //Collection of child graphics.  **private** List<Graphic>childGraphics = **new**ArrayList<Graphic>();  //Prints the graphic.  **publicvoid** print() {  **for** (Graphic graphic : childGraphics) {  graphic.print();  }  }  //Adds the graphic to the composition.  **publicvoid** add(Graphic graphic) {  childGraphics.add(graphic);  }  //Removes the graphic from the composition.  **publicvoid** remove(Graphic graphic) {  childGraphics.remove(graphic);  }  } |

|  |
| --- |
| /\*\* "Leaf" \*/  **class** Ellipse **implements** Graphic {  // Prints the graphic.  **publicvoid** print() {  System.*out*.println("Ellipse");  }  } |

|  |
| --- |
| /\*\* Client \*/  **publicclass** Program {  **publicstaticvoid** main(String[] args) {  // Initialize four ellipses  Ellipse ellipse1 = **new** Ellipse();  Ellipse ellipse2 = **new** Ellipse();  Ellipse ellipse3 = **new** Ellipse();  Ellipse ellipse4 = **new** Ellipse();  // Initialize three composite graphics  CompositeGraphic graphic = **new**CompositeGraphic();  CompositeGraphic graphic1 = **new**CompositeGraphic();  CompositeGraphic graphic2 = **new**CompositeGraphic();  // Composes the graphics  graphic1.add(ellipse1);  graphic1.add(ellipse2);  graphic1.add(ellipse3);  graphic2.add(ellipse4);  graphic.add(graphic1);  graphic.add(graphic2);  // Prints the complete graphic (four times the string "Ellipse").  graphic.print();  }  } |