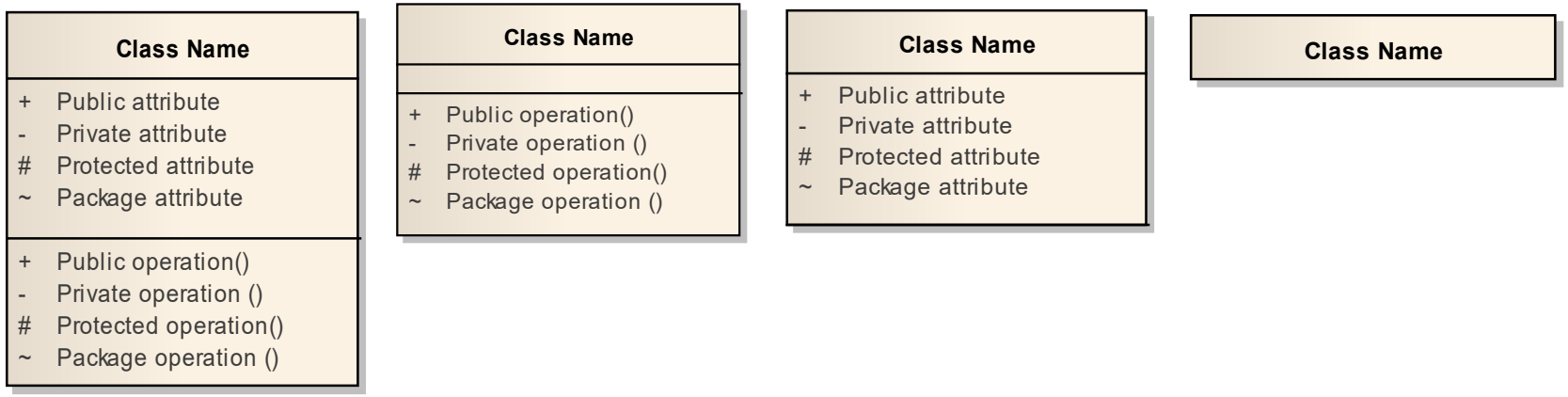
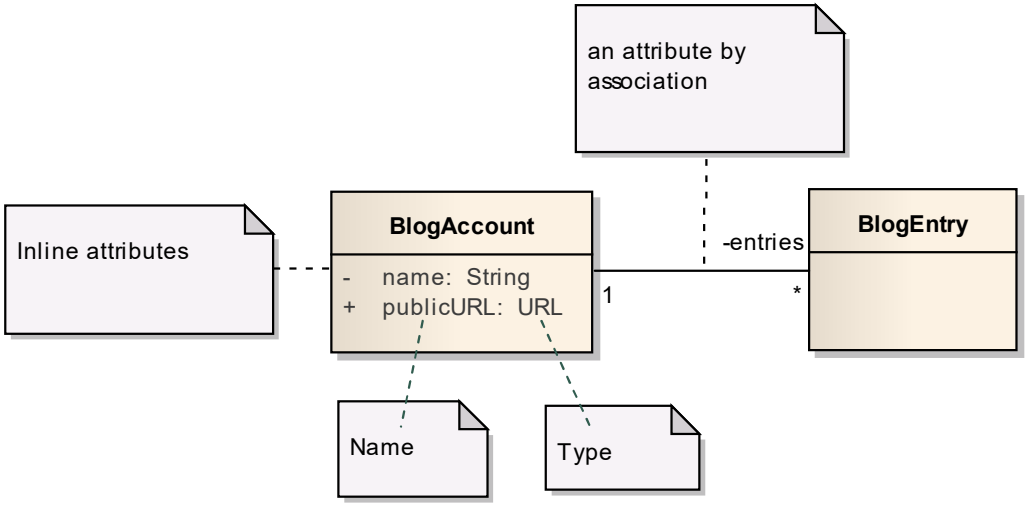
CLASS DIAGRAM

- کلاس توصیف کننده‌ی مجموعه‌ای از object هاست که دارای صفات، عملیات و روابط و semantic مشترکی می‌باشند.
- نمودار کلاس بر ساختار سیستم و جنبه‌های static آن تمرکز دارد و شامل یک مجموعه از کلاس‌ها، واسط‌ها، collaboration ها و روابط بین آن‌ها می‌باشد.

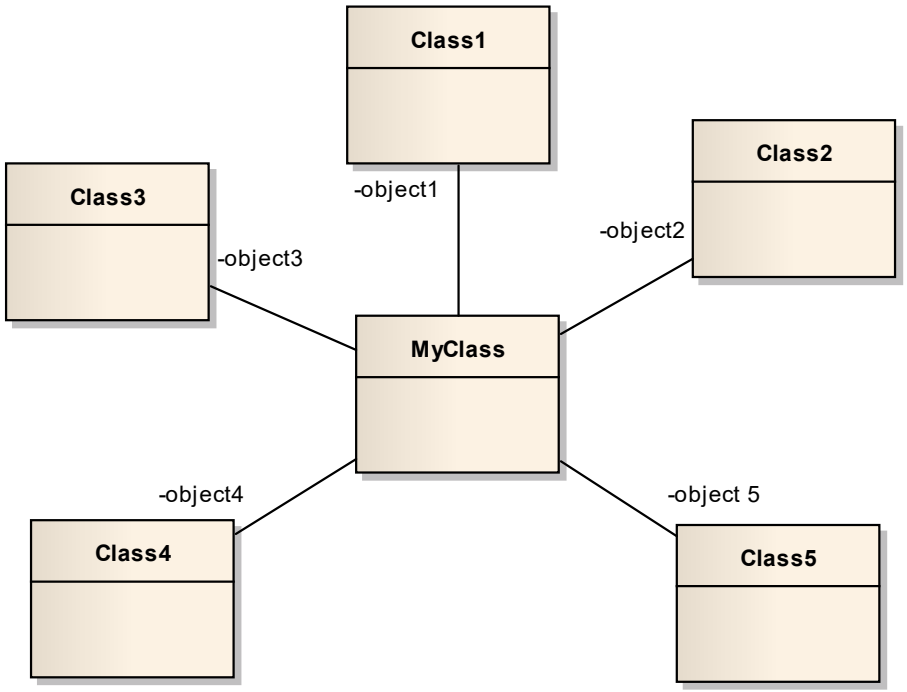


CLASS DIAGRAM

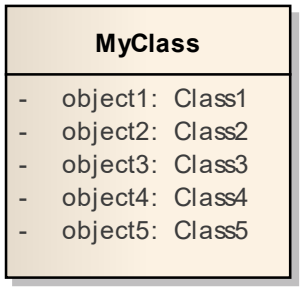
- Attributes
  - بیان کننده‌ی state یک شی هستند و می‌توانند به دو صورت نمایش داده شوند:
    - Inlined: در داخل خود کلاس قرار می گیرند.
    - By relationship: از طریق رابطه association بین کلاس‌ها تعریف می‌شود.



CLASS DIAGRAM

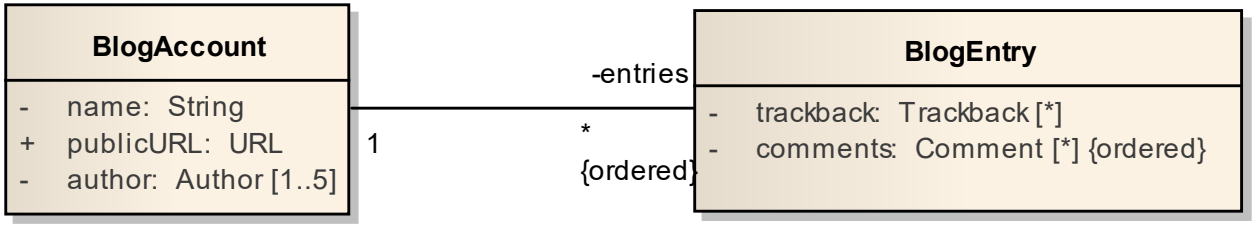


- Attributes

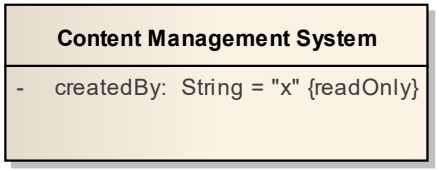


CLASS DIAGRAM

- Attributes
- Multiplicity: در برخی موارد یک attribute به بیش از یک object اشاره دارد.
- Ordered
- Duplicated

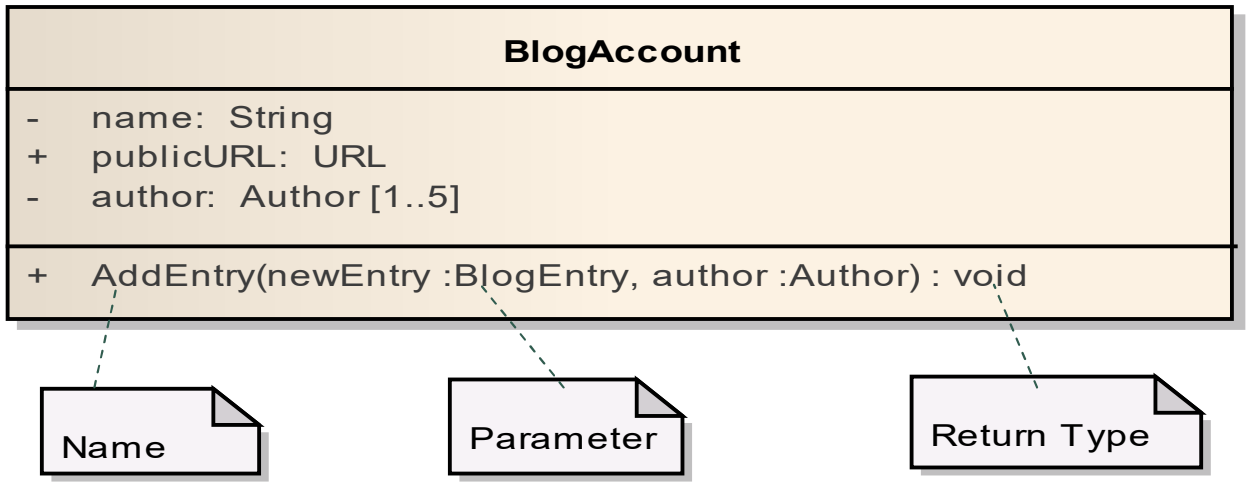


- readonly : مقدار یک attribute پس از این که مقدار اولیه آن set شد قابل تغییر نیست.



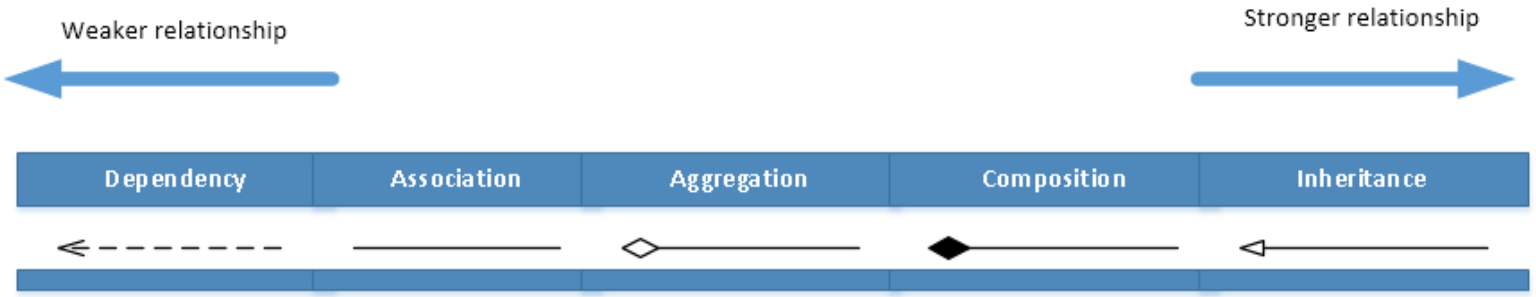
CLASS DIAGRAM

- Operations

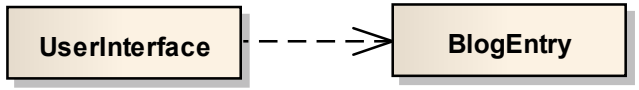


CLASS DIAGRAM

- Class relationship
  - میزان قوی بودن یک ارتباط بر مبنای میزان وابستگی کلاس‌های درگیر در رابطه با یکدیگر مشخص می‌گردد.
  - زمانی که دو کلاس به میزان زیادی به یکدیگر وابسته باشند (tightly coupled) تغییرات یک کلاس به میزان زیادی در کلاس دیگر تاثیر می‌گذارد.



CLASS DIAGRAM

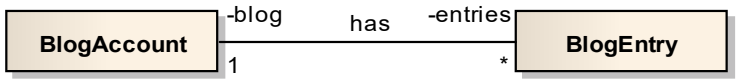


- Dependency
  - ضعیف ترین نوع ارتباط است.
  - در این نوع ارتباط، یک کلاس از کلاس دیگر استفاده می‌کند. این استفاده می‌تواند به صورت‌های گوناگون باشد:
    - یک کلاس به کلاس دیگر پیغام بفرستد. (مثلا استفاده از متدهای کلاس های utility مانند java.math)
    - یک کلاس بخشی از داده‌های یک کلاس دیگر باشد. (برای مثال خروجی یکی از متدهای یک کلاس، از نوع کلاس دیگر باشد.)
    - یک کلاس به صورت آرگومان به یکی از operation های کلاس دیگر فرستاده شود.)
  - یک رابطه کوتاه مدت است. به عبارتی object های یک کلاس با object های کلاس دیگر در تمام life time خود در ارتباط نیستند.

CLASS DIAGRAM

- Association
  - نسبت به dependency قوی تر است.
  - رابطه طولانی مدت، ساختاری و فامیلی بین دو کلاس بقسمی که .Oهای هر یک از این دو کلاس .Oهای کلاس دیگر را در طول عمرش بخاطر می‌آورد.
  - یک کلاس در اثر رابطه association یک reference به کلاس دیگر دارد. (به عبارتی یک attribute از نوع کلاس دیگر دارد که به آن اشاره دارد.)
  - در association، جهت navigation دارای اهمیت است و مشخص می‌کند که کدام کلاس دارای یک reference به کلاس دیگر است.

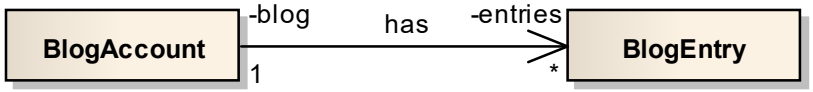
CLASS DIAGRAM



Association

```
public class BlogAccount {

   // Attribute introduced thanks to the association with the BlogEntry class
   private BlogEntry[] entries;

   // ... Other Attributes and Methods declared here ...
}

public class BlogEntry {

   // Attribute introduced thanks to the association with the Blog class
   private BlogAccount blog;

   // ... Other Attributes and Methods declared here ...
}
```

CLASS DIAGRAM



- Association

```
public class BlogAccount {

   // Attribute introduced thanks to the association with the BlogEntry class
   private BlogEntry[] entries;

   // ... Other Attributes and Methods declared here ...
}

public class BlogEntry
{
   // The blog attribute has been removed as it is not necessary for the
   // BlogEntry to know about the BlogAccount that it belongs to.

   // ... Other Attributes and Methods declared here ...
}
```

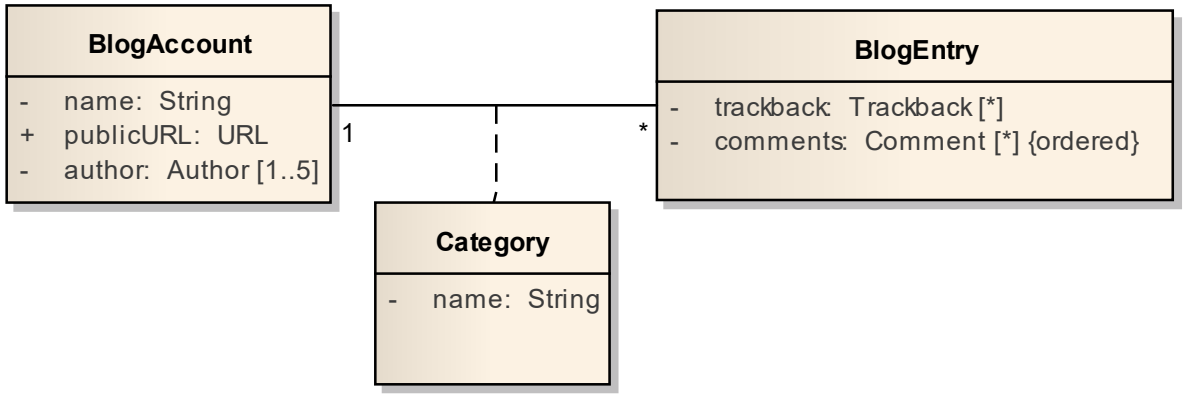
CLASS DIAGRAM

- Association
- Multiplicity

- Exactly one - 1
- Zero or one - 0..1
- Many - 0..* or *
- One or more - 1..*
- Exact Number - e.g. 3..4 or 6
- Or a complex relationship – e.g. 0..1, 3..4, 6..* would mean any number of objects other than 2 or 5

CLASS DIAGRAM

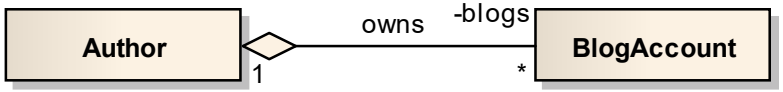
- Association class
  - در برخی موارد رابطه‌ی association موجب معرفی یک کلاس جدید می‌گردد که association class نام دارد.




```
public class BlogAccount {
   private String name;
   private Category[] categories;
   private BlogEntry[] entries;
}

public class Category {
   private String name;
}

public class BlogEntry {
   private String name;
   private Category[] categories
}
```

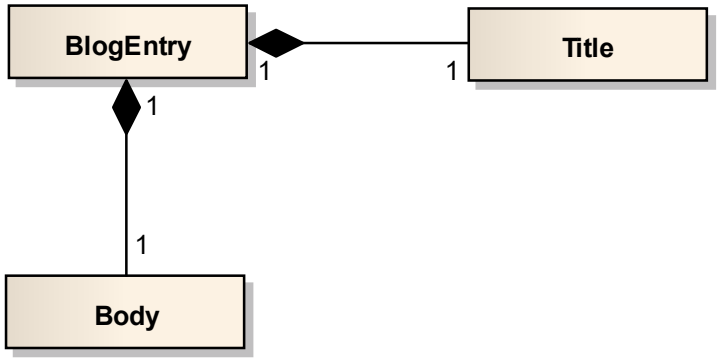
CLASS DIAGRAM

- Aggregation
  - نوع به خصوصی از association است که در آن O های یک کلاس(Whole) شامل O های یک کلاس دیگر(Part) هستند؛ در حالی که این O های کلاس part می‌توانند توسط کلاس‌های دیگر به صورت مشترک مورد استفاده قرار گیرند.

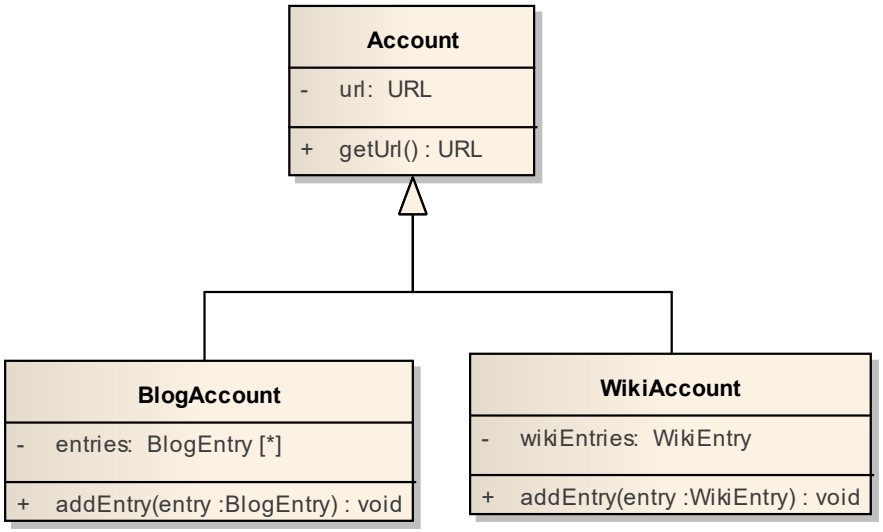


CLASS DIAGRAM

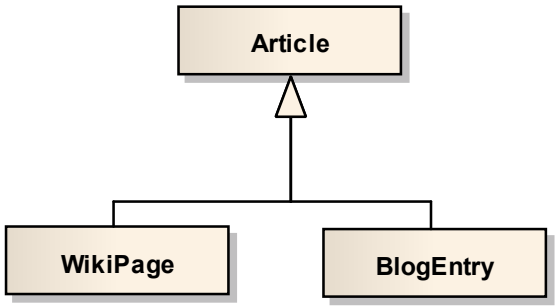
- Composition
  - نوع به خصوصی از association است که در آن O های یک کلاس بخشی از O های کلاس دیگرند. (رابطه part of)
  - object ای که بخشی از object دیگر است دارای Life time یکسان با آن است. (به عبارتی در صورتی که object بزرگتر از بین برود، object جز نیز به همراه آن از بین خواهد رفت.)
  - یک object می‌تواند حداکثر متعلق به یک composition باشد.



CLASS DIAGRAM

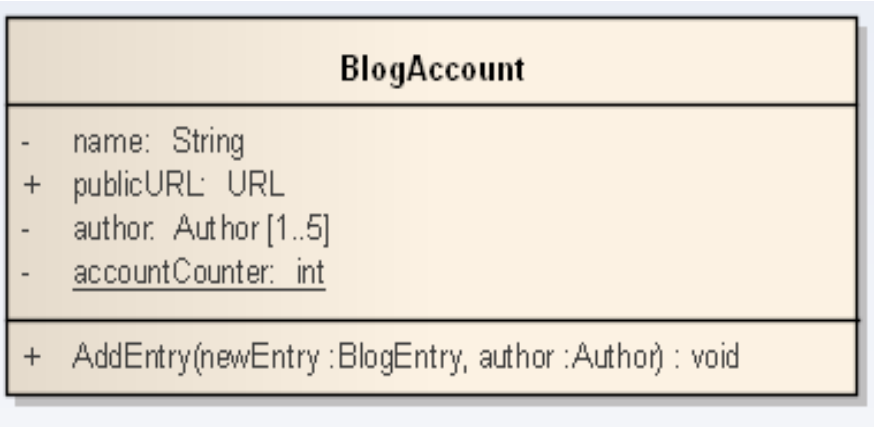


- Inheritance
  - یک کلاس از نوع کلاس دیگر است.

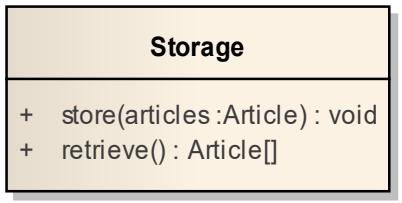


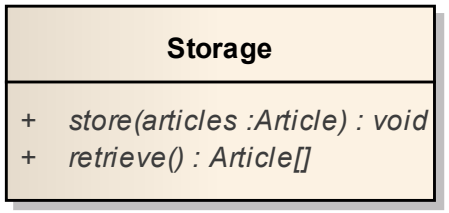
CLASS DIAGRAM

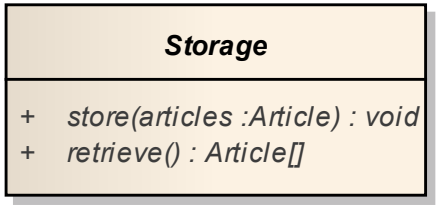
- Static attribute and operation and class
  - Static attributes: attribute هایی که استفاده از آن‌ها فراتر از life time، object های یک کلاس امکان‌پذیر است. به عبارتی این attribute وابسته به خود کلاس هستند نه instance آن. (با استفاده از آن می‌توان یک attribute را بین object های مختلف یک کلاس به اشتراک گذاشت.)
  - staic operation: operation ها معمولا رفتار یک object را بیان می‌کنند، اما با استفاده از staic operation می‌توان رفتار خود کلاس را بیان نمود.

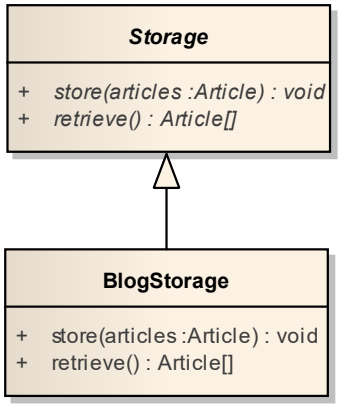


CLASS DIAGRAM





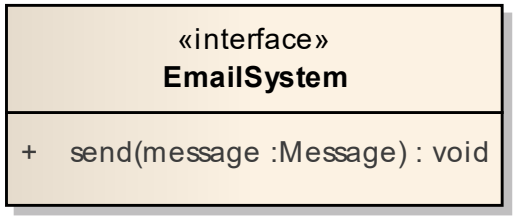




- Abstract class
  - abstarct class کلاس هایی هستند که دارای بخش‌های unimplemented هستند.
  - نمی‌توان آن‌ها را مستقیما instantiate نمود.

CLASS DIAGRAM

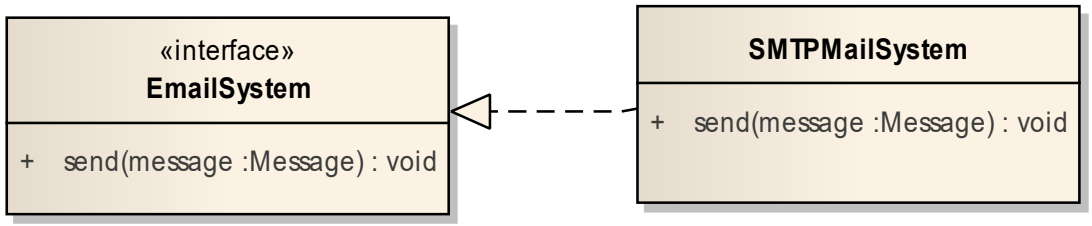
- Interface
  - این امکان را فراهم می‌آورد که بتوانیم تعدادی متد را declare کرده و پیاده‌سازی این متدها را به concrete class ها بسپاریم.
  - اگرچه abstract class نیز می‌تواند این امکان را فراهم کند، اما با توجه به اینکه multiple assignment توسط برخی از زبان‌ها پشتیبانی نمی‌شود، با استفاده از interface می‌توان کلاسی داشت که چندین interface را پیاده‌سازی می‌کند.



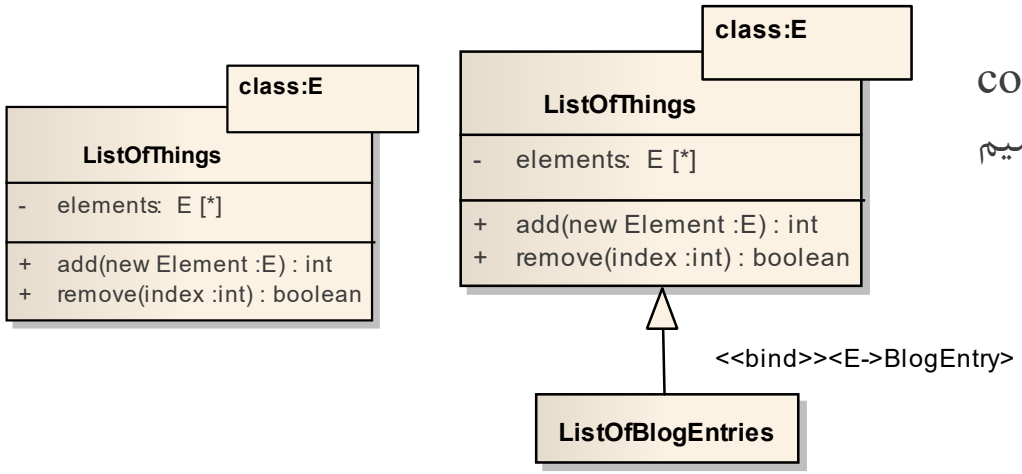

```
public interface EmailSystem {
   public void send(Message message);
}
```

CLASS DIAGRAM

- Interface

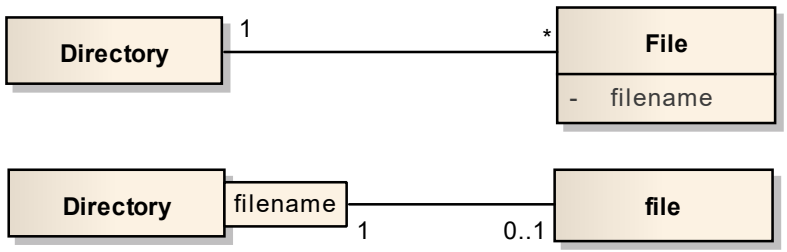


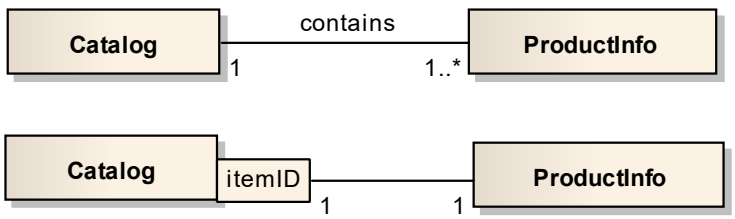
- Template
  - به آن parameterized class نیز گفته می‌شود.
  - زمانی استفاده می‌شود که مشخص نیست concrete class ای که یک کلاس با آن کار میکند چیست و تصمیم گیری در مورد آن به تعویق انداخته شده است.



CLASS DIAGRAM

- Qualification
  - تکنیکی است که با استفاده از **کلیدها** امکان کاهش multiplicity در یک association را فراهم می‌آورد.
  - یکی از روش‌های معمول برای پیاده‌سازی آن hash table است.





CLASS DIAGRAM

```
public class Directory{
  private HashTable files;

  public void addfile (File f) {
     files.put(f.getName(), f);
  }

  public void removeFile(String name) {
     files.remove(name);
  }

  public Account lookupFile(String name) {
     return
       (File) files.get(name);
  }
}
```

- Qualification