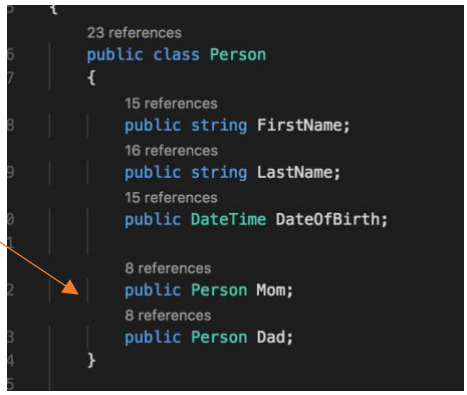
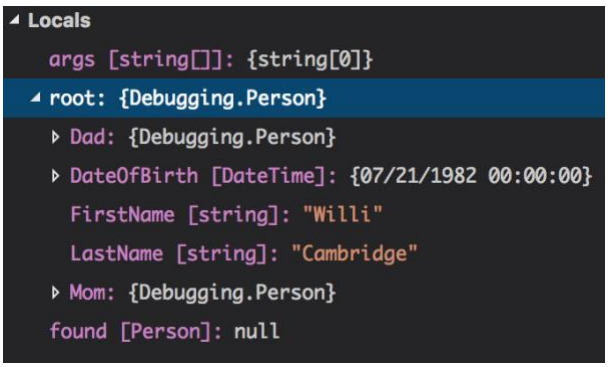
Softwaredesign – Aufgabe 3: Debugging



```
23 references
public class Person
{
    15 references
    public string FirstName;
    16 references
    public string LastName;
    15 references
    public DateTime DateOfBirth;

    8 references
    public Person Mom;
    8 references
    public Person Dad;
}
```

Hier werden die Variablen „Mom“ und „Dad“ initialisiert.

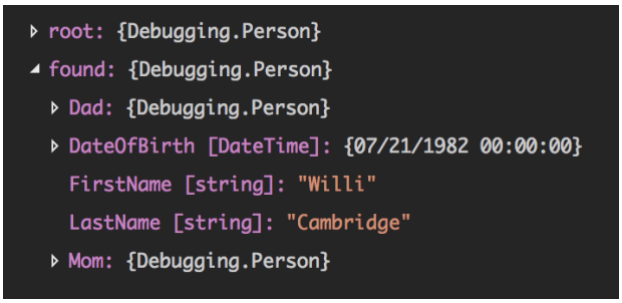


```
Locals
  args [string[]]: {string[0]}
  root: {Debugging.Person}
    Dad: {Debugging.Person}
    DateOfBirth [DateTime]: {07/21/1982 00:00:00}
    FirstName [string]: "Willi"
    LastName [string]: "Cambridge"
    Mom: {Debugging.Person}
  found [Person]: null
```

Beim Anhalten in Z.19 durch einen Breakpoint sehe ich in root, „Willi Cambridge“ und dessen Dad + Mom. Diese Variablen kann ich weiter aufklappe, um mehr Details zu bekommen und dessen Mom + Dad zu sehen.

```
2 references
17  public class Familytree
18  {
        3 references
19      public static Person Find(Person person)
20      {
21          Person ret = null;
22          if (person.LastName != "Battenberg")
23              return person;
24
25          ret = Find(person.Mom);
26          if (ret != null)
27              return ret;
28          ret = Find(person.Dad);
29          return ret;
30      }
```

Hier in der Find-Funktion wird die Bedingung gestellt, ob person.LastName ungleich „Battenberg“ entspricht. Im weiteren wird nach Mom und dann Dad überprüft, in einer Funktion die sich immer selbst aufruft, bis ein Ergebnis gefunden wird.



```
  root: {Debugging.Person}
  found: {Debugging.Person}
    Dad: {Debugging.Person}
    DateOfBirth [DateTime]: {07/21/1982 00:00:00}
    FirstName [string]: "Willi"
    LastName [string]: "Cambridge"
    Mom: {Debugging.Person}
```

Im nächsten Schritt wird die Bedingung erfüllt und „Willi Cambridge“ zurückgegeben. Zudem in der Konsole die„Debugging.Person“ ausgegeben.

```
3 references
19  public static Person Find(Person person)
20  {
21      Person ret = null;
22      if (person.LastName == "Battenberg")
23          return person;
24
25      ret = Find(person.Mom);
26      if (ret != null)
27          return ret;
28      ret = Find(person.Dad);
29      return ret;
30  }
```

Nachdem ich den != Operator auf == geändert habe wurde mir bei Ausführung des Programms eine **NullReferenceException** ausgegeben.

```
MacBook-Pro:L03_Debugging Tim$ dotnet run

Unhandled Exception: System.NullReferenceException: Object reference not set to an instance of an object.
   at Debugging.Familytree.Find(Person person) in /Users/Tim/Documents/GitHub/SoftwareDesign/Class/L03_Debugging/Fami
lyTree.cs:line 22
   at Debugging.Familytree.Find(Person person) in /Users/Tim/Documents/GitHub/SoftwareDesign/Class/L03_Debugging/Fami
lyTree.cs:line 25
   at Debugging.Familytree.Find(Person person) in /Users/Tim/Documents/GitHub/SoftwareDesign/Class/L03_Debugging/Fami
lyTree.cs:line 25
   at Debugging.Familytree.Find(Person person) in /Users/Tim/Documents/GitHub/SoftwareDesign/Class/L03_Debugging/Fami
lyTree.cs:line 25
   at Debugging.Familytree.Find(Person person) in /Users/Tim/Documents/GitHub/SoftwareDesign/Class/L03_Debugging/Fami
lyTree.cs:line 25
```

```
20      {
21          Person ret = null;
22          if (person.LastName == "Battenberg")
23              return person;
24
25          ret = Find(person.Mom);
26          if (ret != null)
27              return ret;
28          ret = Find(person.Dad);
29          return ret;
30      }
```

Im nächsten Schritt habe ich einen Breakpoint in Z. 25 gesetzt, um den Fehler via Debugging zu finden.

```
$exception [NullReferenceException]: {System.NullReferenceException: O…
  Data [IDictionary]: {System.Collections.ListDictionaryInternal}
  HResult [int]: -2147467261
  HelpLink [string]: null
  InnerException [Exception]: null
  Message [string]: "Object reference not set to an instance of an obj…
  Source [string]: "Debugging"
  StackTrace [string]: "   at Debugging.Familytree.Find(Person person)…
  TargetSite [MethodBase]: {Debugging.Person Find(Debugging.Person)}
  Static members
  Non-Public members
person [Person]: null
ret [Person]: null
```

Erst ist die Person „Willi Camrbidge“ als rootperson angegeben und durch weitere Prozedurschritte verändert sich die Person zu Diana Spencer, Franzi Roche und Ruth Grill bis schießlich die NullReferenceException geworfen wird. Dies passiert, da bei Ruth Grill keine „Mom“ gefunden wird, da diese nicht deklariert ist.

```
3 references
public static Person Find(Person person)
{
    Person ret = null;
    if (person.LastName == "Battenberg")
        return person;

    if (person.Mom != null)
        ret = Find(person.Mom);
    if (ret != null)
        return ret;

    if (person.Dad != null)
        ret = Find(person.Dad);
    return ret;
}
```

Um die NullReferenceException zu umgehen, habe ich eine einfache if-Abfrage gemacht, um zu prüfen ob person.Mom/Dad null ist. Wenn nicht, dann wird die Funktion ausgeführt, und somit wird keine NullReferenceException mehr ausgeworfen.

Die komplexere Bedingung:

```
34
        2 references
35      public static Person FindPersonByAge(Person person, int min_age, int max_age)
36      {
37          Person ret = null;
38          int age = DateTime.Now.Year - person.DateOfBirth.Year;
39          if (age >= min_age && age <= max_age)
40              return person;
41          if(person.Mom != null)
42              ret = FindPersonByAge(person.Mom, min_age, max_age);
43          if (ret != null)
44              return ret;
45          if(person.Dad != null)
46              ret = FindPersonByAge(person.Dad, min_age, max_age);
47          return ret;
48      }
```

Hier wird nun die erste Person ausgegeben, wessen Alter zwischen den neuen Parametern (min_age und max_age) liegt .