

## Protokoll zu Aufgabe 4 Debugging:

Die Datenstruktur Person ist rekursiv, d.h. sie besteht aus verketteten Objekten, also den Knoten. ( Mom und Dad). Diese Datenstrukturen sind dynamisch, weil zur Laufzeit des Programmes neue Knoten erzeugt und verkettet werden können.

Methode BuildTree(): baut einen Beispiel-Baum auf. Breakpoint in Zeile 19, Inhalt von root:

```
Locals
  args [string[]]: {string[0]}
  root: {Aufgabe_4.Person}
    Dad: {Aufgabe_4.Person}
    DateOfBirth [DateTime]: {21.07.1982 00:00:00}
    FirstName [string]: "Willi"
    LastName [string]: "Cambridge"
    Mom: {Aufgabe_4.Person}
    found [Person]: null
```

Die Methode BuildTree() mit der Variable root und dem Typen Person enthält die Objekte Dad, Mom und die Variable DateOfBirth. In DateOfBirth sind FirstName, sowie LastName enthalten. Die Methode wird aufgerufen und überprüft die Bedingung: if (person.LastName != "Battenberg") Die erste Person, die die Bedingung erfüllt, heißt: „Willi Cambridge“.

**VARIABLES**

**Locals**

- \$exception [NullReferenceException]: {System.NullRef...}
- person [Person]: null
- ret [Person]: null

**WATCH**

**CALL STACK** (PAUSED ON EXCEPTION)

- Aufgabe 4.dll!Aufgabe\_4.Familytree.Find(Aufgabe\_4.Perso
- Aufgabe 4.dll!Aufgabe\_4.Familytree.Find(Aufgabe\_4.Perso
- Aufgabe 4.dll!Aufgabe\_4.Familytree.Find(Aufgabe\_4.Perso
- Aufgabe 4.dll!Aufgabe\_4.Familytree.Find(Aufgabe\_4.Perso
- Aufgabe 4.dll!Aufgabe\_4.Familytree.Find(Aufgabe\_4.Perso
- Aufgabe 4.dll!Aufgabe\_4.Program.Main(string[] args) Lin

**Code Editor:**

```
18 {
19     3 references
20     public static Person Find(Person person)
21     {
22         Person ret = null;
23         if (person.LastName == "Battenberg")
```

**Exception Message:** An unhandled exception of type 'System.NullReferenceException' occurred in Aufgabe 4.dll: 'Object reference not set to an instance of an object.'

**Stack Trace:**

- at Aufgabe\_4.Familytree.Find(Person person) in C:\Users\User\Documents\GitHub\Softwaredesign\Aufgabe 4\FamilyTree.cs:line 22
- at Aufgabe\_4.Familytree.Find(Person person) in C:\Users\User\Documents\GitHub\Softwaredesign\Aufgabe 4\FamilyTree.cs:line 25
- at Aufgabe\_4.Familytree.Find(Person person) in C:\Users\User\Documents\GitHub\Softwaredesign\Aufgabe 4\FamilyTree.cs:line 25
- at Aufgabe\_4.Familytree.Find(Person person) in C:\Users\User\Documents\GitHub\Softwaredesign\Aufgabe 4\FamilyTree.cs:line 25
- at Aufgabe\_4.Familytree.Find(Person person) in C:\Users\User\Documents\GitHub\Softwaredesign\Aufgabe 4\FamilyTree.cs:line 25
- at Aufgabe\_4.Familytree.Find(Person person) in C:\Users\User\Documents\GitHub\Softwaredesign\Aufgabe 4\FamilyTree.cs:line 25
- at Aufgabe\_4.Program.Main(String[] args) in C:\Users\User\Documents\GitHub\Softwaredesign\Aufgabe 4\Program.cs:line 19

**Code Editor (continued):**

```
23         return person;
24     }
25     ret = Find(person.Mom);
```

**DEBUG CONSOLE:**

S. Module is optimized and the debugger option 'Just my code' is enabled.

Loaded 'C:\Program Files\dotnet\shared\Microsoft.NETCore.App\2.0.3\System.Runtime.InteropServices.dll'

Wir haben die erste if-Bedingung zu if (person.LastName == "Battenberg") geändert, sodass nicht gleich die erste Person („Willi“) zurückgegeben wird. Dies wirft aber eine Exception auf (siehe Screenshot).

Als erste Person wird Willi Cambridge überprüft, danach durchläuft das Programm in Zeile 32 Person Mom und überprüft „Diana“, „Franzi“ und „Ruth“. Da der Nachname „Battenberg“ bei Person Mom nicht definiert ist, wird eine Exception angezeigt.

