Legacy C Code to Dataflow

Visualizing different aspects of legacy code for code comprehension or even automatically generated documentation of parts of the code is part of the research interest in the Visible Code project. One part of it is visualizing legacy imperative code by abstracting the behavior into a visual representation of the flow of data through methods in the code and the use/origin of specific data artifacts.

This page is for documenting some technical aspects and examples on how the "C to Dataflow" processor turns C code into SCCharts dataflow.

Schema of generated SCCharts

```
scchart schema {
        input: only input variables visible from the outside in references of this state, e.g. method
parameters, read/written variables in other states (if, while, etc.). All these variables are suffixed with
"_in".
        output: only output variables visible from the outside in references of this state, e.g. the method
return value, written variables/arrays in other states (if, while, etc.). All these variables are suffixed
with "_out".
        No other variables are declared here.

        dataflow {
                - declarations of local variables
                - SSA variables of local and input variables
                - reference declarations of contained states

                equations:
                - connecting all in-/outputs of references
                - other connections that may be specific to some construct (e.g. if/while conditions,
combination of possible return values, etc.)
                - assignment of all last SSA forms of variables to their _out variables of the surrounding
state
        }
}

scchart possibleReferencedState {
        - same as above. To be serializable as a texutal SCChart again, all referenced SCCharts are added to
the root SCCharts object.
}
```

Example code and their generated SCCharts

Following are some example C methods and their translations in SCCharts Dataflow.

```
int zero() {
        return 0;
}

int main(int x) {
        int a = zero();
        return a;
}
```

```
scchart zero {
        output int _res_out
label "_res"

        dataflow DF-zero "zero"
{
                @Hide
                int _res_0

                _res_0 = 0

                _res_out =
_res_0
        }
}
scchart main {
        input int x_in label "x"
        output int _res_out
label "_res"

        dataflow DF-main "main"
{
                @hide
                ref zero zero
                @Hide
                int a_0 label
"a"
                @Hide
                int _res_0

                a_0 = zero.
_res_out

                _res_0 = a_0

                _res_out =
_res_0
        }
}
```

```
int main(int x) {
        int a;
        if (x > 42) {
                a = 1337 * x;
        } else {
                a = 420 / x;
        }
        return a;
}
```

```
scchart if_0 {
        input int a_in label "a"
        input int x_in label "x"
        output int a_out label
"a"

        dataflow "if_0" {
                bool
_conditional
                ref if_0then
if_0then
                ref if_0else
if_0else
                @Hide
                int then_0a
label "then_0a"
                @Hide
                int else_0a
label "else_0a"

                _conditional =
x_in > 42

                if_0then.a_in =
a_in

                if_0then.x_in =
```

```
x_in

                if_0else.a_in =
a_in

                if_0else.x_in =
x_in

                then_0a =
if_0then.a_out

                else_0a =
if_0else.a_out

                a_out =
_conditional ? then_0a : else_0a
        }
}
scchart if_0then {
        input int a_in label "a"
        input int x_in label "x"
        output int a_out label
"a"

        dataflow DF-if_0then
"then" {
                @Hide
                int a_0

                a_0 = 1337 *
x_in

                a_out = a_0
        }
}
scchart if_0else {
        input int a_in label "a"
        input int x_in label "x"
        output int a_out label
"a"

        dataflow DF-if_0else
"else" {
                @Hide
                int a_0

                a_0 = 420 / x_in

                a_out = a_0
        }
}
scchart main {
        input int x_in label "x"
        output int _res_out
label "_res"

        dataflow DF-main "main"
{
                @Hide
                int a_0 label
"a", a_1, a_2
                ref if_0 if_0
                @Hide
                int _res_0

                if_0.a_in = a_0

                if_0.x_in = x_in

                a_1 = if_0.a_out
```

```
                a_2 = if_0.a_out

                _res_0 = a_2

                _res_out =
_res_0
        }
}
```