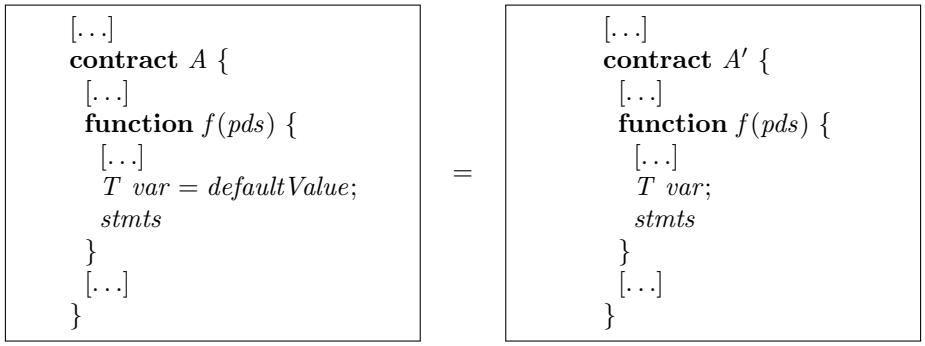
**Rule 0.9** ⟨*Avoid Explicit Zero Initialization*⟩




**where**

*var* is a variable of type $T$ being declared and initialized;

$T$ is the type of the variable (e.g., **uint**, **bool**, **address**, etc.);

*defaultValue* is the default zero value for type $T$ (e.g., 0 for **uint**, *false* for **bool**, **address**(0) for **address**);

*stmts* represents the sequence of statements following the declaration;

*pds* are the parameter declarations of function $f$.

**provided**

The explicit initialization value is exactly the default zero value for type $T$;

Solidity automatically initializes variables to their default zero values;

The variable *var* can be either a local variable or a state variable;

Removing the explicit initialization does not affect program semantics.

**Invariant:**

Let $s_i$ and $s_i'$ be the initial state of $A$ and $A'$, respectively.

Let $s_f$ and $s_f'$ be the state reached by $A$ and $A'$, respectively, after $A.f()$ and $A'.f()$ are executed from $s_i$ and $s_i'$, respectively.

Then, the coupling invariant is

$$\forall\, s_i, s_i' \;.\; (s_i = s_i') \rightarrow (s_f = s_f')$$