RTF Views generated by EiffelStudio IDE

Cluster View

```
class ACCOUNT General
        cluster: model
        description:
                "A bank account with deposit and withdraw
                operations. A bank account may not have a negative balance."
        create: new
Ancestors
        ANY
Queries
        balance: VALUE
        is_equal (other: ACCOUNT): BOOLEAN
        name: STRING
Commands
        deposit (v: VALUE)
        withdraw (v: VALUE)
Constraints
        balance non negative
```

Contract View (Short)

```
note
        description: "[
                A bank account with deposit and withdraw
                operations. A bank account may not have a negative balance.
        ]"

class interface ACCOUNT create
        new

feature -- queries
        name: STRING
        balance: VALUE

feature -- Commands
        deposit (v: VALUE)
                require
                        positive: v > v.zero
                ensure
                        correct_balance: balance = old balance + v

        withdraw (v: VALUE)
                require
                        v > v.zero
                        balance - v >= v.zero
                ensure
                        correct_balance: balance = old balance - v

feature -- equality
        is_equal (other: like Current): BOOLEAN
                        -- Is other value equal to current
                ensure then
                        Result = (name ~ other.name and balance = other.balance)
invariant
        balance_non_negative: balance >= balance.zero

end -- class ACCOUNT
```

Text View (Short)

```
note
	description: "[
		A bank account with deposit and withdraw
		operations. A bank account may not have a negative balance.
	]"

class ACCOUNT inherit
	ANY redefine is_equal end
create
	new

feature {NONE} -- create
	new (a_name: STRING_8)
			-- create an account for a_name with zero balance
		do
			create name.make_from_string (a_name)
		ensure
			created: name ~ a_name
			balance_zero: balance = balance.zero
		end

feature -- Queries
	name: STRING
	balance: VALUE

feature -- Commands
	deposit (v: VALUE)
		require
			positive: v > v.zero
		do
			balance := balance + v
		ensure
			correct_balance: balance = old balance + v
		end

	withdraw (v: VALUE)
		require
			positive: v > v.zero
				balance - v >= v.zero
		do
			balance := balance - v
		ensure
			correct_balance: balance = old balance - v
		end

feature -- Queries of Comparison

	is_equal (other: like Current): BOOLEAN
			-- Is other value equal to current
		do
			Result := name ~ other.name and balance = other.balance
		ensure then
				Result = (name ~ other.name and balance = other.balance)
		end

invariant
	balance_non_negative: balance >= balance.zero

end -- class ACCOUNT
```