```
──────────────── MODULE VolSyncPopulator ────────────────
Model for the "data populator" functionality in VolSync.

The goal with this model is to verify that the populator controller can properly interact with
the ReplicationDestination controller to provision PVCs based on the LatestImage that has been
saved.
EXTENDS FiniteSets, TLC
CONSTANTS OIDs,    A symmetry set representing object UUIDs
          nil

The allowable values for an object's kind
ObjKinds ≜ {"RD", "VS", "PV", "PVC"}


--fair algorithm populator
variables
    Objs = {}   The set of objects in the system

define
    TypeOk ≜ ∧ ∀ o ∈ Objs : ∧ ∀ f ∈ {        Objects must have at least:
                                   "k",       * kind
                                   "oid"      * UUID
                                 } : f ∈ DOMAIN o
                              ∧ o.k ∈ ObjKinds          All objects have a valid kind
                              ∧ o.oid ∈ OIDs            All objects have a valid ID
                              ∧ ∀ x ∈ Objs : ∨ x = o    IDs are unique
                                             ∨ x.oid ≠ o.oid

     FreeOIDs is the current set of unused IDs
    FreeOIDs ≜ OIDs \ {o.oid : o ∈ Objs}
     Empty record
    empty ≜ [x ∈ {} ↦ {}]
end define

process User ∈ {"user"}
begin
UStart:
    with i ∈ FreeOIDs do
        Objs := Objs ∪ {[oid ↦ i, k ↦ "RD", LI ↦ nil]};
    end with ;
end process

process RDController ∈ {"RDC"}
begin
RDBegin:
    with rd ∈ {o ∈ Objs : o.k = "RD"}, i ∈ FreeOIDs do
        Objs := (Objs \ {rd}) ∪ {[oid ↦ i, k ↦ "VS"], [rd EXCEPT !.LI = i]};
    end with ;
    print (Objs);
end process
end algorithm



 BEGIN TRANSLATION (chksum(pcal) = “60ba6cfc” ∧ chksum(tla) = “582f3ad5”)
VARIABLES Objs, pc

 define statement
TypeOk ≜ ∧ ∀ o ∈ Objs   : ∧ ∀ f ∈ {
                                  “k”,
                                  “oid”
                               } : f ∈ DOMAIN o
                             ∧ o.k ∈ ObjKinds
                             ∧ o.oid ∈ OIDs
                             ∧ ∀ x ∈ Objs : ∨ x = o
                                            ∨ x.oid ≠ o.oid


FreeOIDs ≜ OIDs \ {o.oid : o ∈ Objs}

empty ≜ [x ∈ {} ↦ {}]


vars ≜ ⟨Objs, pc⟩

ProcSet ≜ ({“user”}) ∪ ({“RDC”})

Init ≜  Global variables
        ∧ Objs = {}
        ∧ pc = [self ∈ ProcSet ↦ CASE self ∈ {“user”} → “UStart”
                                 □   self ∈ {“RDC”} → “RDBegin”]

UStart(self) ≜ ∧ pc[self] = “UStart”
               ∧ ∃ i ∈ FreeOIDs :
                    Objs′ = (Objs ∪ {[oid ↦ i, k ↦ “RD”, LI ↦ nil]})
               ∧ pc′ = [pc EXCEPT ![self] = “Done”]

User(self) ≜ UStart(self)

RDBegin(self) ≜ ∧ pc[self] = “RDBegin”
                ∧ ∃ rd ∈ {o ∈ Objs : o.k = “RD”} :
                     ∃ i ∈ FreeOIDs :
                       Objs′ = ((Objs \ {rd}) ∪ {[oid ↦ i, k ↦ “VS”], [rd EXCEPT !.LI = i]})
                ∧ PrintT((Objs′))
                ∧ pc′ = [pc EXCEPT ![self] = “Done”]

RDController(self) ≜ RDBegin(self)
```

```
 Allow infinite stuttering to prevent deadlock on termination.
Terminating ≜ ∧ ∀ self ∈ ProcSet : pc[self] = "Done"
              ∧ UNCHANGED vars

Next ≜ (∃ self ∈ {"user"} : User(self))
          ∨ (∃ self ∈ {"RDC"} : RDController(self))
          ∨ Terminating

Spec ≜ ∧ Init ∧ □[Next]_vars
       ∧ WF_vars(Next)

Termination ≜ ◇(∀ self ∈ ProcSet : pc[self] = "Done")

 END TRANSLATION
├──────────────────────────────────────────────────────────────────────────┤

└──────────────────────────────────────────────────────────────────────────┘
\* Modification History
\* Last modified Fri Feb 03 11:42:33 EST 2023 by jstrunk
\* Created Fri Feb 03 09:54:49 EST 2023 by jstrunk
```