1 Data Model

As an example, let's define two structures, one to describe/list of "Activities" (like a term project, course project, etc.) and a second one to describe/list the enrolled students (assuming that each enrolled student has one, or more, advisors and a set of reviewers).

> ***Note:*** As in any "procedural language", one is advised to carefully design the data model, since this will shape the functions which will set and use said data.
>
> ***Note:*** Pay attention to the use of the tildes, ~ , since those definitions will be made, most likely, in an *expl3* code régime, one has to remember that spaces are ignored, therefore, if needed, one has to explicitly use a tilde instead of a space.

1.1 Activity Set

For the activities one could set an "starray" as follow:

```
\starray_new:n {activity}
\starray_def_from_keyval:nn {activity} {
    name = Activity's~ name ,
    acronym = ACRO ,
    coord . struct =  {
        name = Coordinator's~ name,
        title = Coordinator's~ title ,
      } ,
    calendar . struct = {
        date = {-day-} ,
        week = {-week-} ,
        event = {-event-} ,
      } ,
    chkID = ,         %%% 'unique ID' for checklists
    chkmarked = ,     %%% This shall be a prop list of   marked itens
    chkunmarked = ,   %%% This shall be a prop list of unmarked itens
    chkref = ,        %%% This shall be a prop list of ref      itens
  }
```

Whereas, the "coord" sub-structure is for the activity's coordinator, whilst "calendar" shall (for instance) contains a list of calendar events, and, finally, the many "chk* " will be used for a "check list".

> ***Note:*** The "chkID" (and checklists). In many cases it's handy to have an unique identifier for a given structure. That can be obtained with `\starray_get_unique_ID:nN`, and to avoid calling this function time and time again, one can just store that ID as a field for later use.
>
> ***Note:*** Could the Coordinator's name and title be a direct property (dismissing the "coord" sub-structure)? of course, that's a matter of taste/choice, on how to model it.

1.2 Student Set

Similarly, a student's structure might contain, besides student's name, work title, some flags, an advisor (and co-advisor, if needed), reviewer's list (with a provision for reviewer's grade, if needed).

Of course, one doesn't need to define a *starray* structure using `\starray_def_from_keyval:nn`, but, as in this, if the set of properties is known, it always makes for a cleaner definition.

> ***Note:*** The fields/properties defaults can be anything, including usual LaTeX 2ε commands, like a `\rule` which is handy, for instance, when generating forms, e.g., if the fields are all set, a form can be created with the proper values, otherwise, it will be created with "rules" in place (no need to test if the properties were set).

```
\starray_new:n {student}
\starray_def_from_keyval:nn {student} {
  self = , %% this shall be self hash (if any)
  first = ,
  last = ,
  name = \rule{\l__stdemo_name_rule_dim}{.1pt} ,
  ID    = \rule{\l__stdemo_ID_rule_dim}{.1pt} ,
  email = \rule{\l__stdemo_email_rule_dim}{.1pt} ,
  worktitle = \rule{\l__stdemo_worktitle_rule_dim}{.1pt} ,
  remarks = ,
  board-local = {~local/place~} ,
  board-date   = {~date~} ,
  board-time  = {~time~} ,
  gradeavrg = 0,
  grade = ,
  flag-null = \c_false_bool , %% IF no grade was given
  flag-graded = \c_false_bool , %%% IF gradeavrg AND finalgrade already calculated (or defined)
  flag-approved = \c_false_bool ,
  flag-coadvisor = \c_false_bool ,
  advisor . struct = {
    first = ,
    last =  ,
    name = \rule{\l__stdemo_name_rule_dim}{.1pt},
    institution = \rule{\l__stdemo_name_rule_dim}{.1pt},
    title = \rule{\l__stdemo_title_rule_dim}{.1pt} ,
    email = \rule{\l__stdemo_email_rule_dim}{.1pt} ,
  } ,
  coadvisor . struct = {
    first = ,
    last =  ,
    name = \rule{\l__stdemo_name_rule_dim}{.1pt},
    institution = \rule{\l__stdemo_name_rule_dim}{.1pt},
    title = \rule{\l__stdemo_title_rule_dim}{.1pt} ,
    email = \rule{\l__stdemo_email_rule_dim}{.1pt} ,
  } ,
  reviewer . struct = {
    first = ,
    last =  ,
    name = \rule{\l__stdemo_name_rule_dim}{.1pt},
    institution = \rule{\l__stdemo_name_rule_dim}{.1pt},
    title = \rule{\l__stdemo_title_rule_dim}{.1pt} ,
    email = \rule{\l__stdemo_email_rule_dim}{.1pt} ,
    pointA = ,
    pointB = ,
    pointC = ,
    pointD = ,
    grade = 0 ,
    flag-set = \c_false_bool ,
  } ,
 }
```

2 Auxiliary Functions

Once the data layout is set (see 1) the next step is to define a set of (document level) functions, so the data can be initialized and used by the end user.

2.1 Activity Functions

One could define a single function to initialize all fields (using a key=val interface), but, in a more traditional approach one can set two functions to start the initialization process `\NewActivity` and `\ActivitySet`.

```
\tl_new:N \l__stdemo_tmpID_tl
\NewDocumentCommand{\NewActivity}{m} {
    \starray_new_term:nn {activity}{#1}
    \starray_new_term:nn {activity.coord}{}
    \starray_get_unique_id:nNTF {activity} \l__stdemo_tmpID_tl
      {}
      {}
    \starray_gset_prop:nnV {activity}{chkID} \l__stdemo_tmpID_tl
    \prop_new_linked:c {l__stdemo_ \l__stdemo_tmpID_tl _chkmarked_prop}
    \prop_new_linked:c {l__stdemo_ \l__stdemo_tmpID_tl _chkunmarked_prop}
    \prop_new_linked:c {l__stdemo_ \l__stdemo_tmpID_tl _chkref_prop}
}

\NewDocumentCommand{\ActivitySet}{O{}mm} {
  \tl_if_blank:nTF {#1}
    {
      \starray_set_prop:nnn {activity}{name}{#3}
      \starray_set_prop:nnn {activity}{acronym}{#2}
    }
    {
      \starray_set_prop:nnn {activity[#1]}{name}{#3}
      \starray_set_prop:nnn {activity[#1]}{acronym}{#2}
    }
}
```

The idea is to (normally) use one right after the other, though, once created with `\NewActivity`, an activity can be initialized/changed at a later point using the optional argument from `\ActivitySet`.

> ***Note:*** Every time a *starray* is instantiated, up to two hashes are created: a numerical one (starting at one) and an "user defined one". In the `\NewActivity` function above, the given argument is that hash, so the just created instance can be later referenced by it. Of course, it must be an unique ID/hash.
>
> ***Note:*** One thing to be noticed about *starrays*: every structure has an associated internal index (iterator). When you create a new instance, this iterator always points to the newly created one, therefore, sparing the use of an explicit index.

Similarly, one can define some functions to set the activity's coordinator. Of course, it's up to the package programmer to choose if one, two (or more) functions for this.

```
\NewDocumentCommand{\ActivitySetCoord}{O{}mO{}}{
  \tl_if_blank:nTF {#1}
    {
      \starray_gset_prop:nnn {activity.coord}{name}{#2}
    }
    {
      \starray_gset_prop:nnn {activity[#1].coord}{name}{#2}
    }
}

\NewDocumentCommand{\ActivitySetCoordTitle}{O{}m} {
  \tl_if_blank:nTF {#1}
    { \starray_set_prop:nnn {activity.coord}{title}{#2} }
    { \starray_set_prop:nnn {activity[#1].coord}{title}{#2} }
}
```

And the associated "Calendar Events", assuming there will be a fixed set of events (each semester, each year), leaving the date to be set later on. `\ActivitySetNewEvent` second parameter is a hash to reference said event (to be used, for instance, with `\ActivitySetEventDay`), and the third parameter is it description.

```
\NewDocumentCommand{\ActivitySetNewEvent}{O{}mm}{
  \tl_if_blank:nTF {#1}
    {
      \starray_new_term:nn {activity.calendar}{#2}
      \starray_gset_prop:nnn {activity.calendar}{event}{#3}
    }
    {
      \starray_new_term:nn {activity[#1].calendar}{#2}
      \starray_gset_prop:nnn {activity[#1].calendar}{event}{#3}
    }
}

\NewDocumentCommand{\ActivitySetEventDay}{O{}mmm}{
  \tl_if_blank:nTF {#1}
    {
      \starray_gset_from_keyval:nn {activity.calendar[#2]}
        {
            date = {#3} ,
            week = {#4} ,
        }
    }
    {
      \starray_gset_from_keyval:nn {activity[#1].calendar[#2]}
        {
            date = {#3} ,
            week = {#4} ,
        }
    }
}
```

```
\NewDocumentCommand{\ActivitySelect}{m}
  {
    \starray_set_iter_from_hash:nn {activity}{#1}
  }
```

```
\NewDocumentCommand{\Activity}{O{}m}{
  \tl_if_blank:nTF {#1}
    { \starray_get_prop:nn {activity}{#2} }
    { \starray_get_prop:nn {activity[#1]}{#2} }
}

\NewDocumentCommand{\ActivityCoord}{O{}m}{
  \tl_if_blank:nTF {#1}
    { \starray_get_prop:nn {activity.coord}{#2} }
    { \starray_get_prop:nn {activity[#1].coord}{#2} }
}
```

```
\NewDocumentCommand{\ActivityCalendarIterate}{m}{
    \starray_iterate_over:nn{activity.calendar}{#1}
}
```