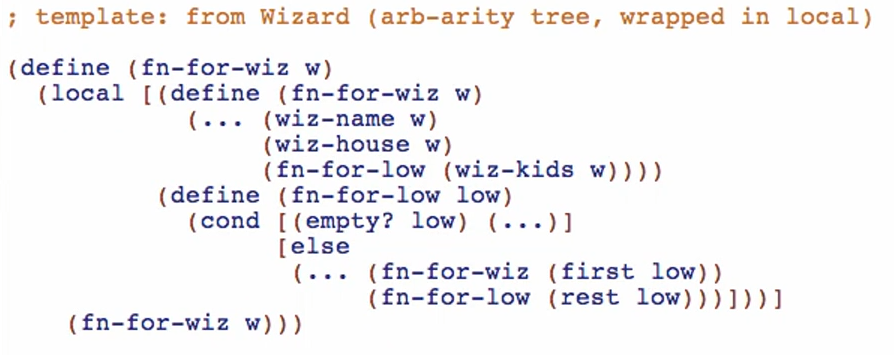


Function definition (Comment out the first function we did earlier)

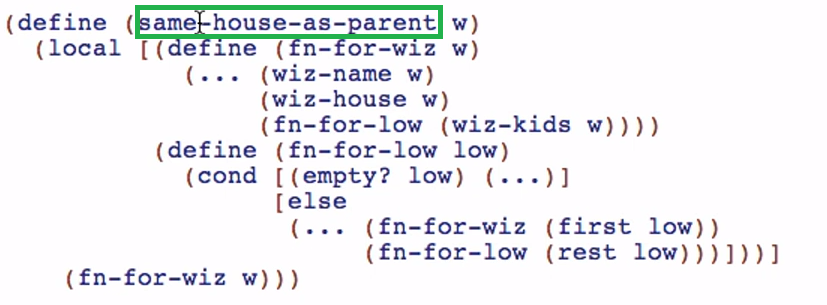
**Template**

1. Original structural recursion/mutual recursion
2. Add accumulator and initial value
3. Work through the accumulator

Copy first template from Wizard

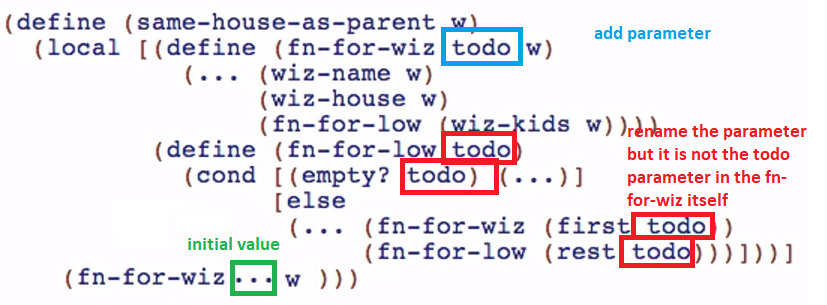


Rename the outer call



Add the worklist accumulator, rename the parameter for fn-for-low, and add … for initial value of todo

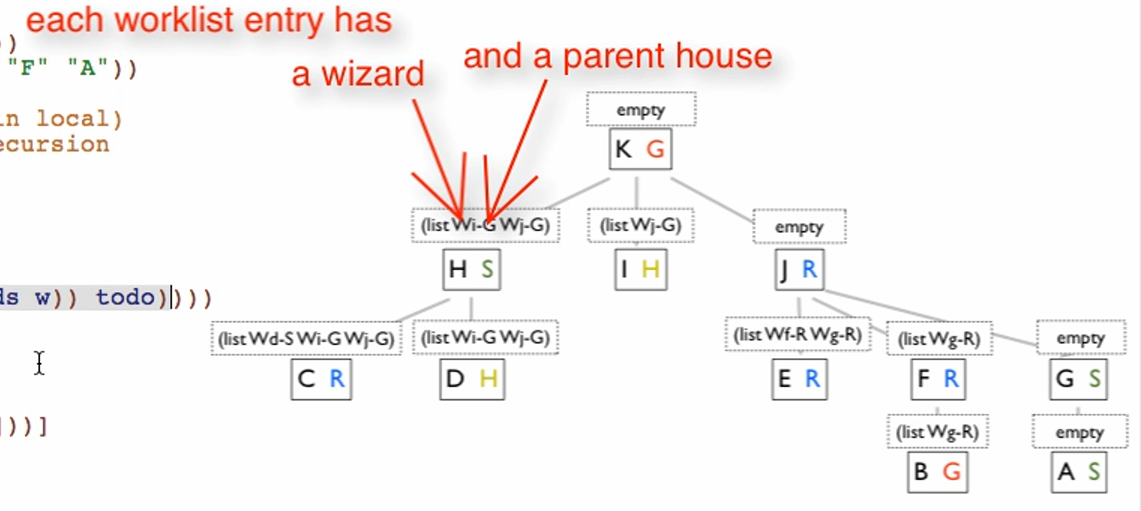




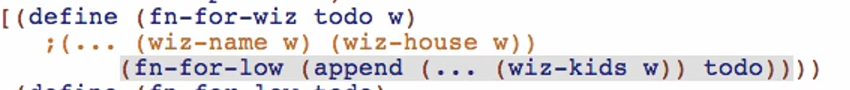
Add the appending of wiz-kids to todo



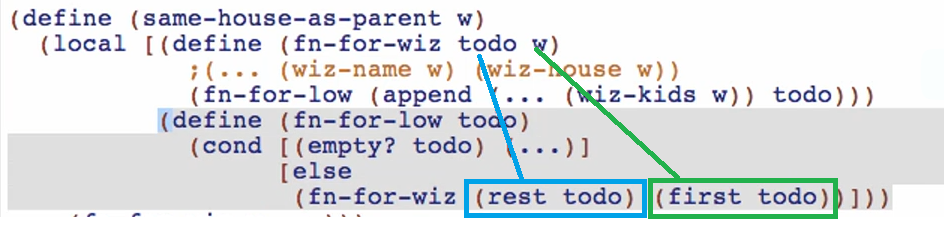
Doing … before wiz-kids call because our accumulator now will be more complicated



Make fn-for-wiz tail recursive



Make fn-for-low tail recursive



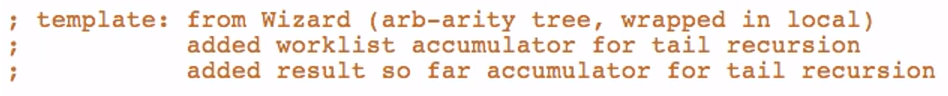
Working through the todo accumulator

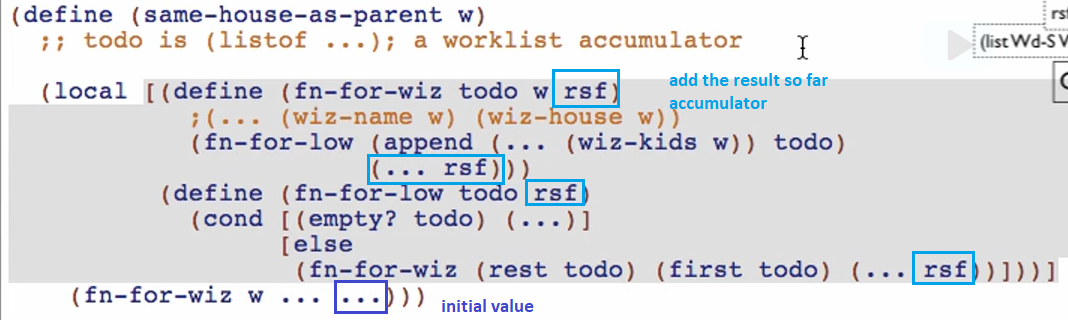
*Signature and purpose*



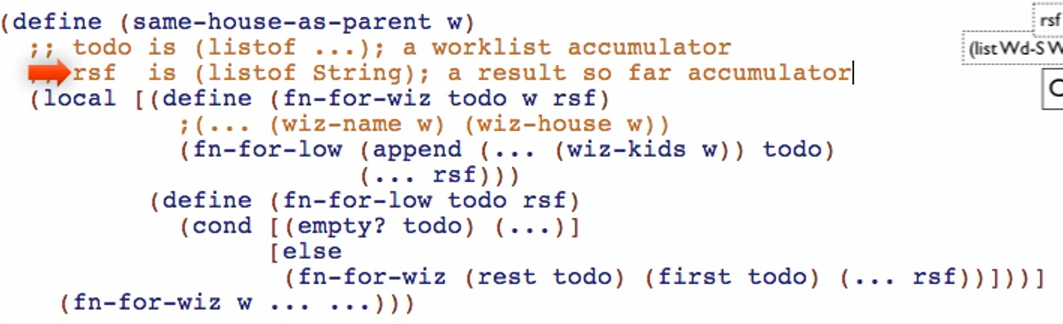
*Example* – see our picture in Worklist Accumulators 2 – Part 3

Adding the result so far accumulator (but in the worklist accumulator)





Working through the rsf accumulator

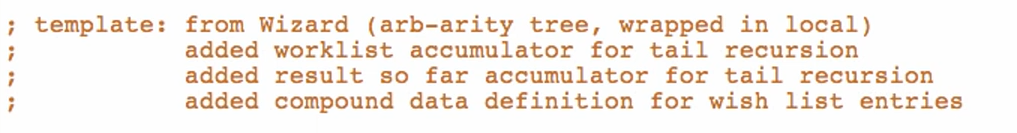
*Signature & purpose* 

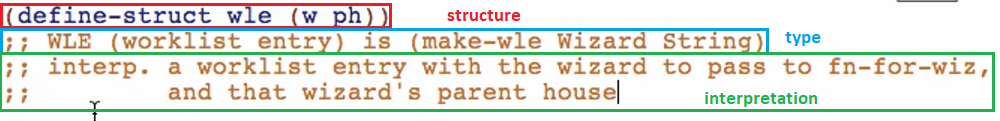
But because we need the worklist entry to have a wizard and a parent house



We need to have a compound data entry

Defining local **define-struct** for our compound data entry

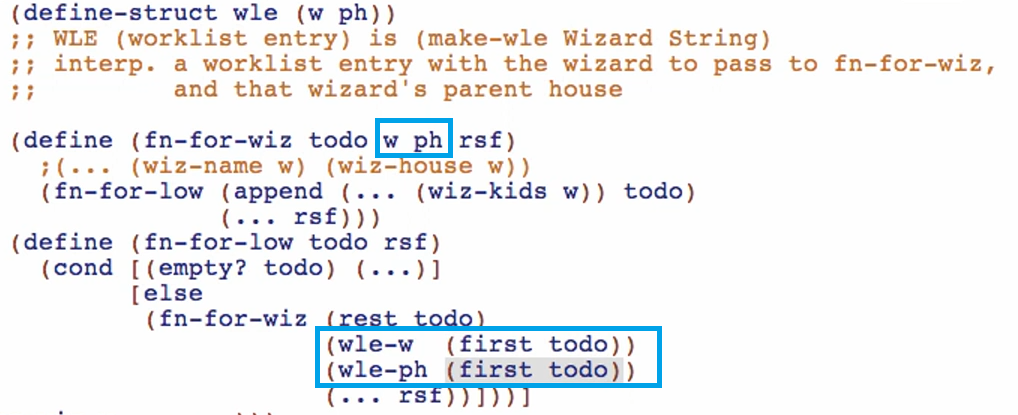


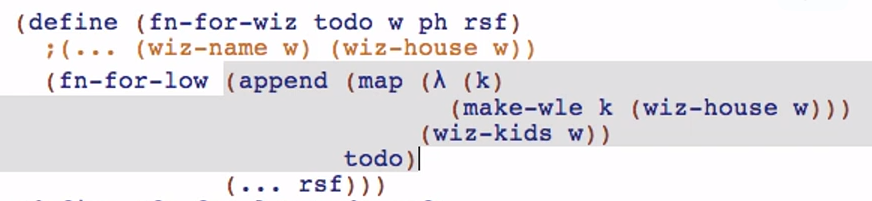


So we also need a “ph” parameter for fn-for-wiz

* A context preserved INSIDE the worklist accumulator entries, not really a context for the whole

This is like unpacking of the w and ph for the worklist entry



Then this is the packing up of w and ph for the worklist entry. On the (… (wiz-kids w)) todo) 

This is just like the normal mapping in JS or Java!

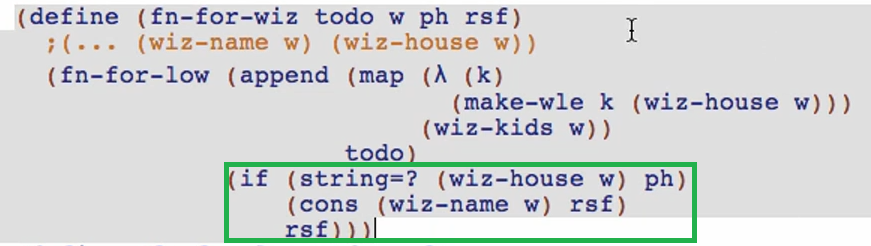
Providing the initial value for the todo accumulator (worklist accumulator)



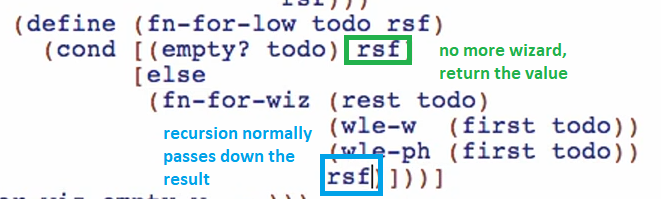
Working through the rsf accumulator

fn-for-wiz

Condition to add to the result so far: wiz-house of the current must be same as ph



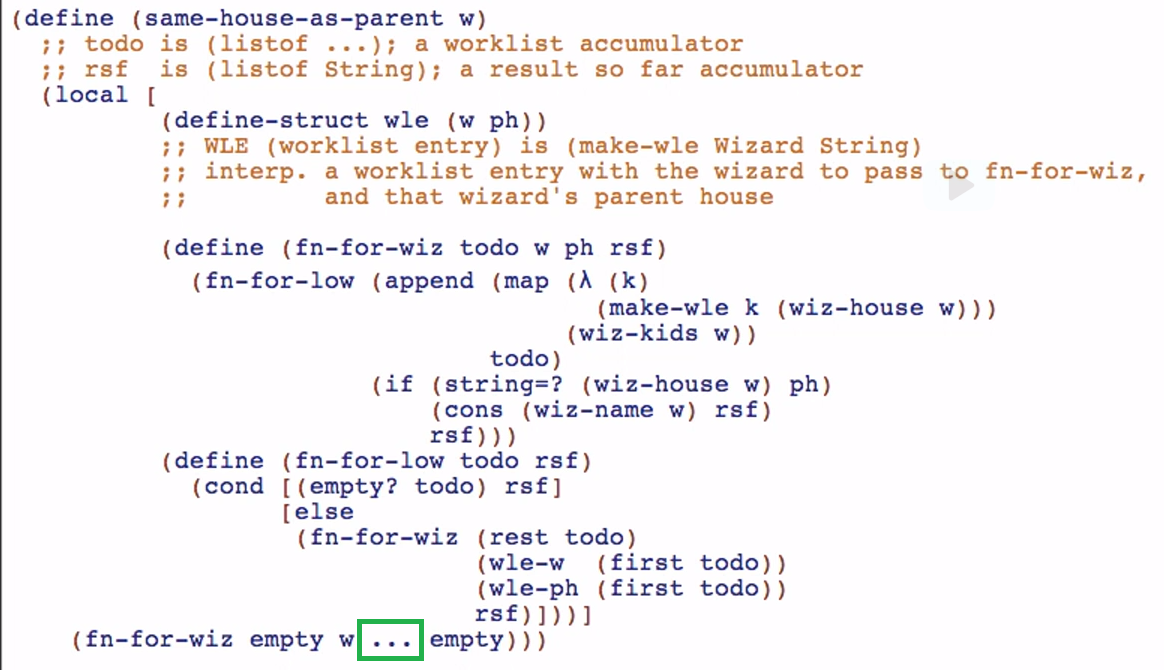
fn-for-low



Initial value



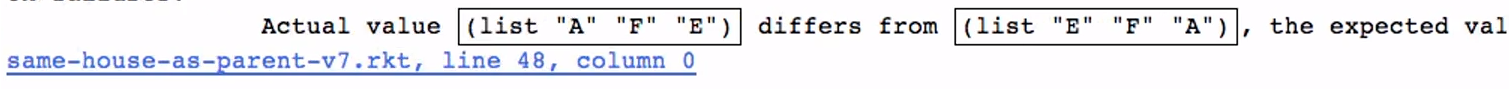
Put dots for ph parameter in fn-for-wiz



And the initial value for ph (the parent of the root of the wizard tree), is just “”



Run and debug



This is our error because the traversal of tail recursive function is not the same as the non-tail recursive one. It has the same elements but different order

We can just flip the check-expect result but make sure that it is based on the traversal we drawn earlier in our picture/domain analysis



Our structure:

