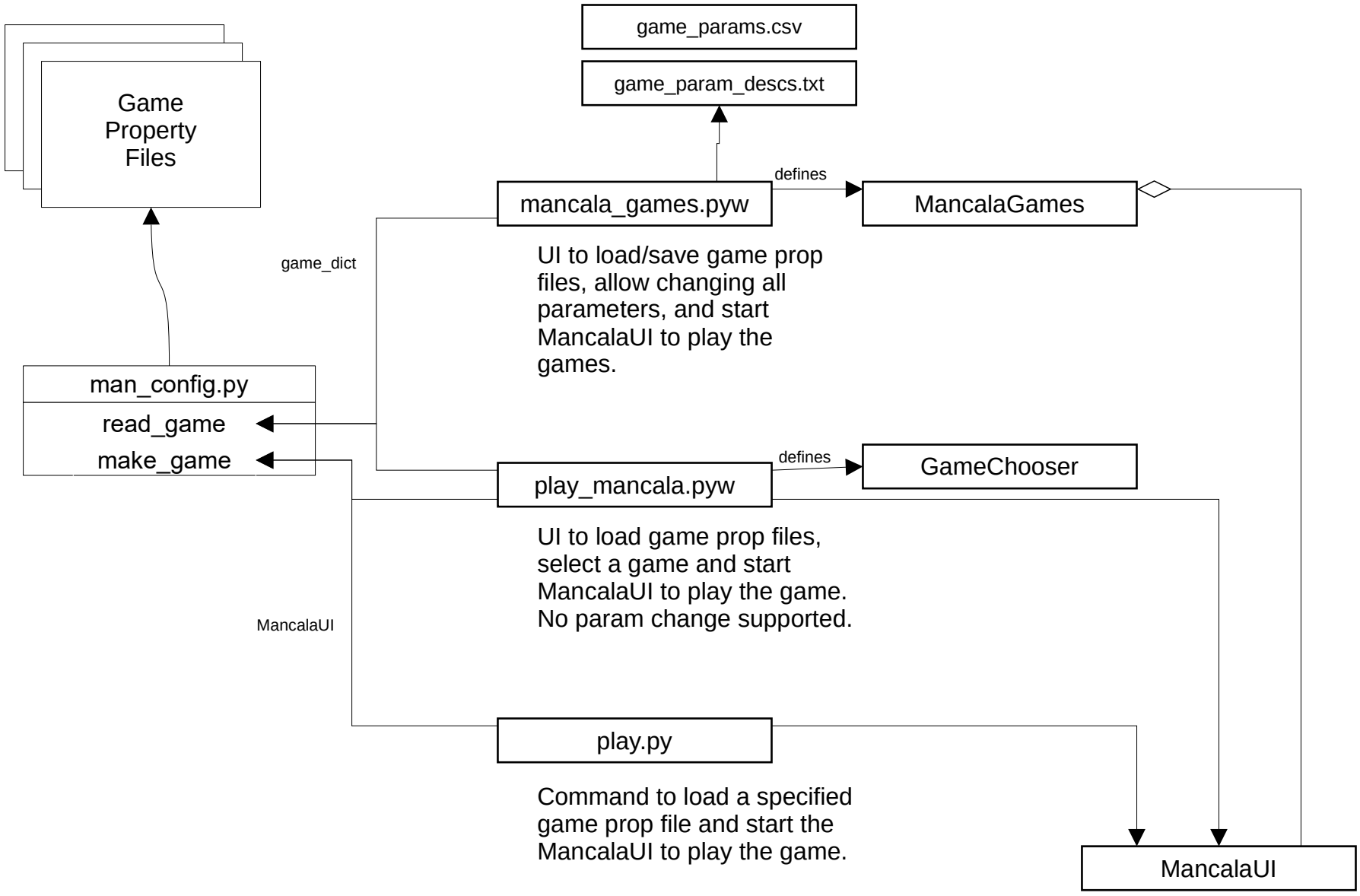
Mancala Games

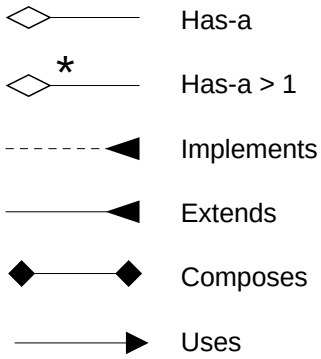


Notation Conventions

Class Diagram Conventions

Abstract Base Class

Primarily Data



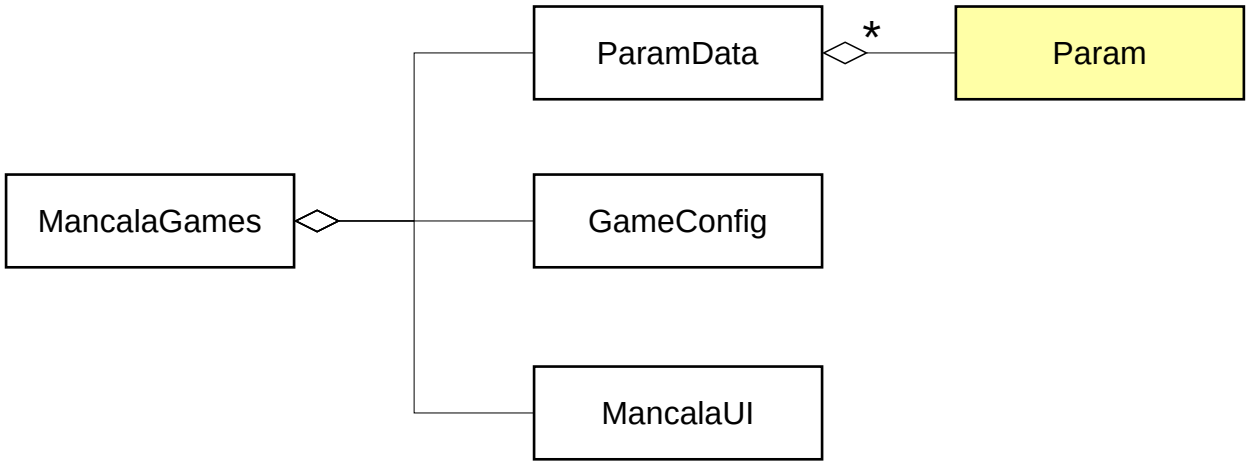
Deco Chain Conventions

- One path down the deco chain is used.
- Intersecting arrows are decision points.
- Shown in **call order** from start dot  (constructed in reverse order).
- Calls down the deco chain maybe at any point in each deco's processing.
- Some deco's do not call down the deco chain even if there is a follow-on deco.
- All paths shown might not be possible (see ginfo_rules).

Optional deco

Deco Chain in Seperate Diagram

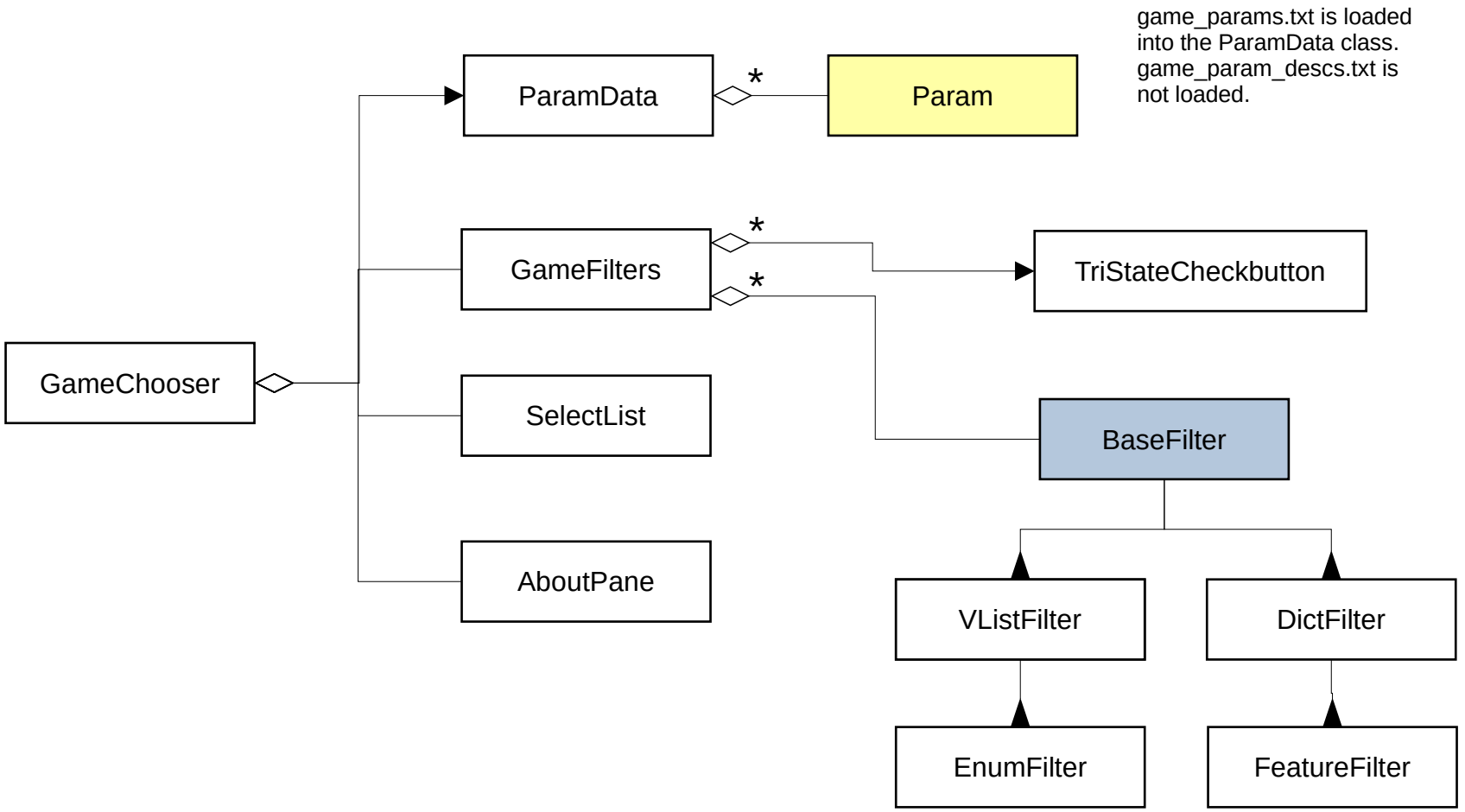
MancalaGames
(the Mancala Games UI class)



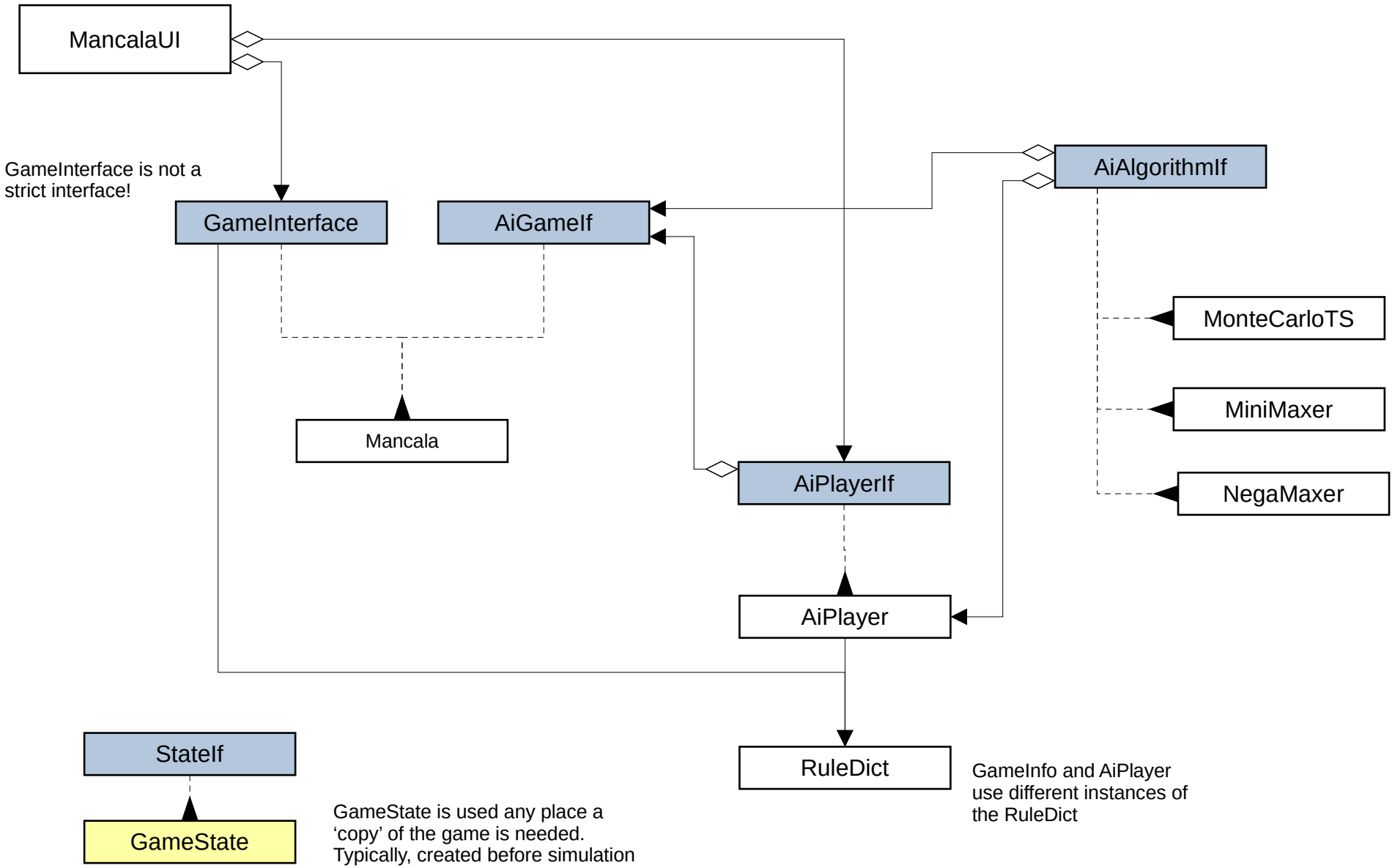
game_params.txt and game_param_descs.txt are loaded into the ParamData class.

Management of the game configuration file is done with the GameConfig class in MancalaUI

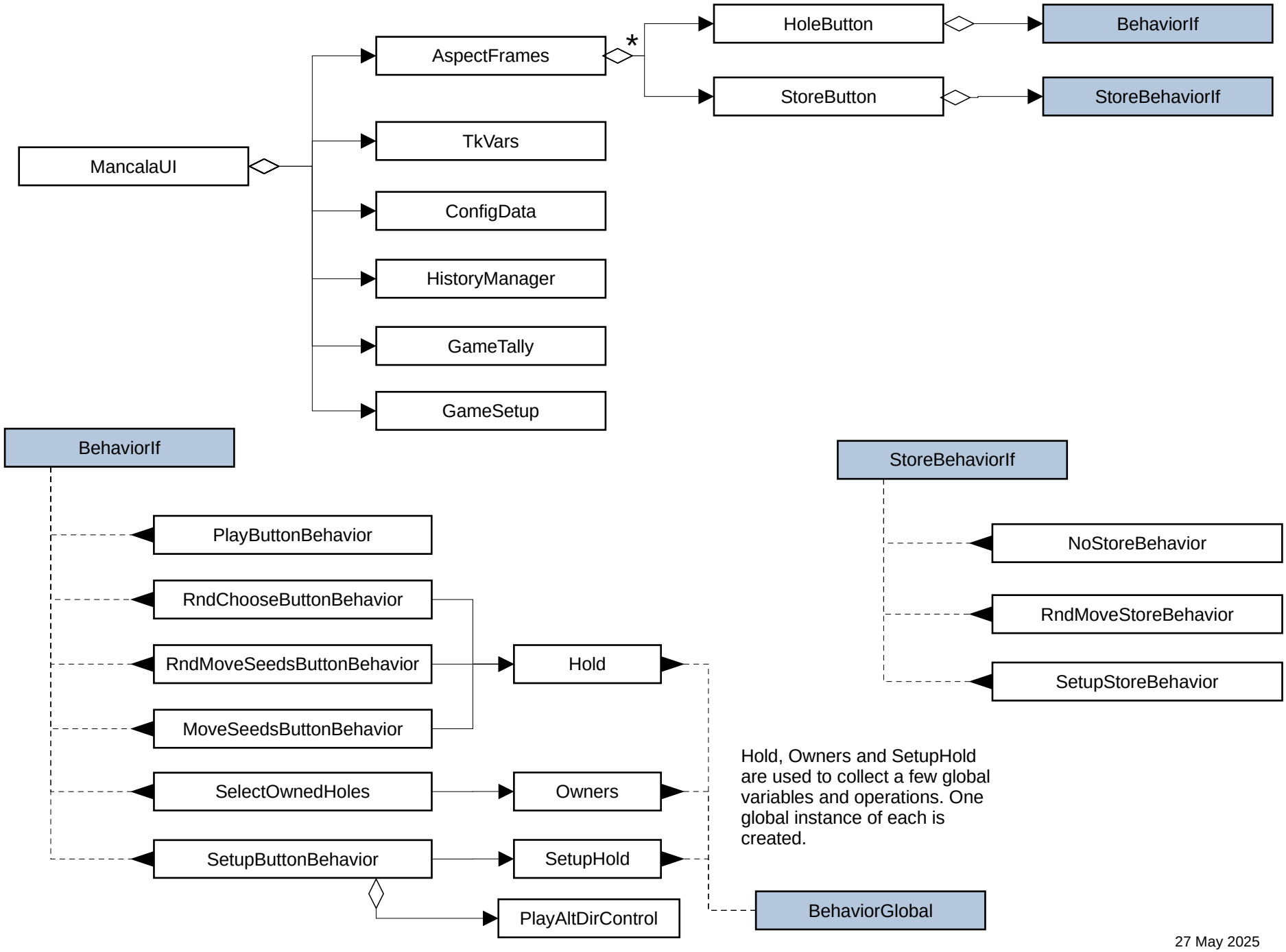
GameChooser
(the Play Mancala UI class)



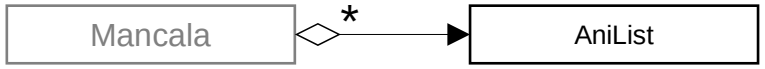
Mancala, GameState, AIPlayer and AIAlgorithm



Mancala UI Classes



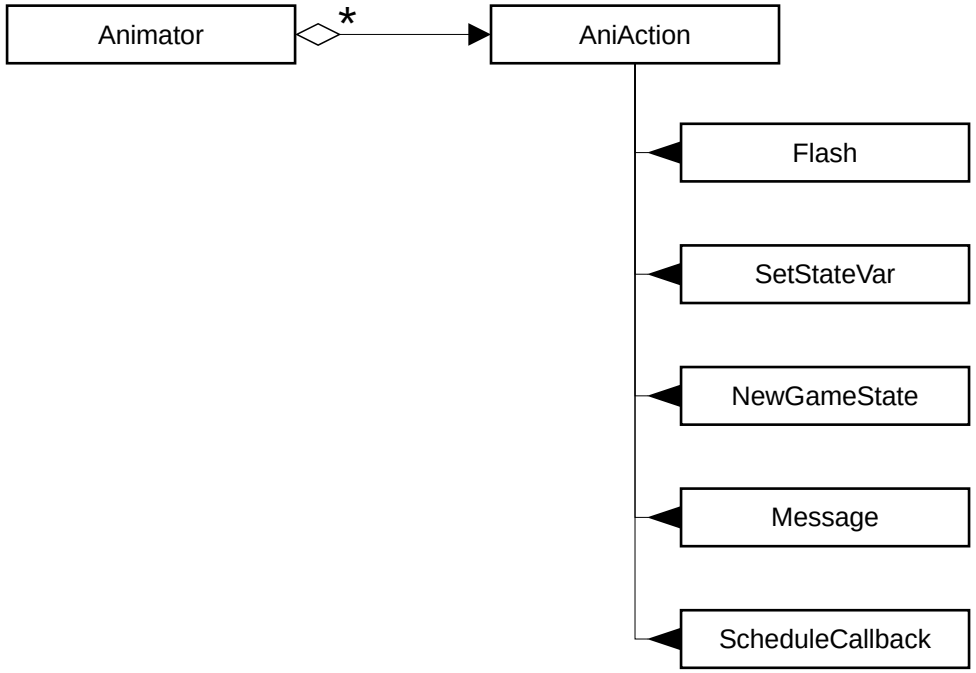
Animator Classes



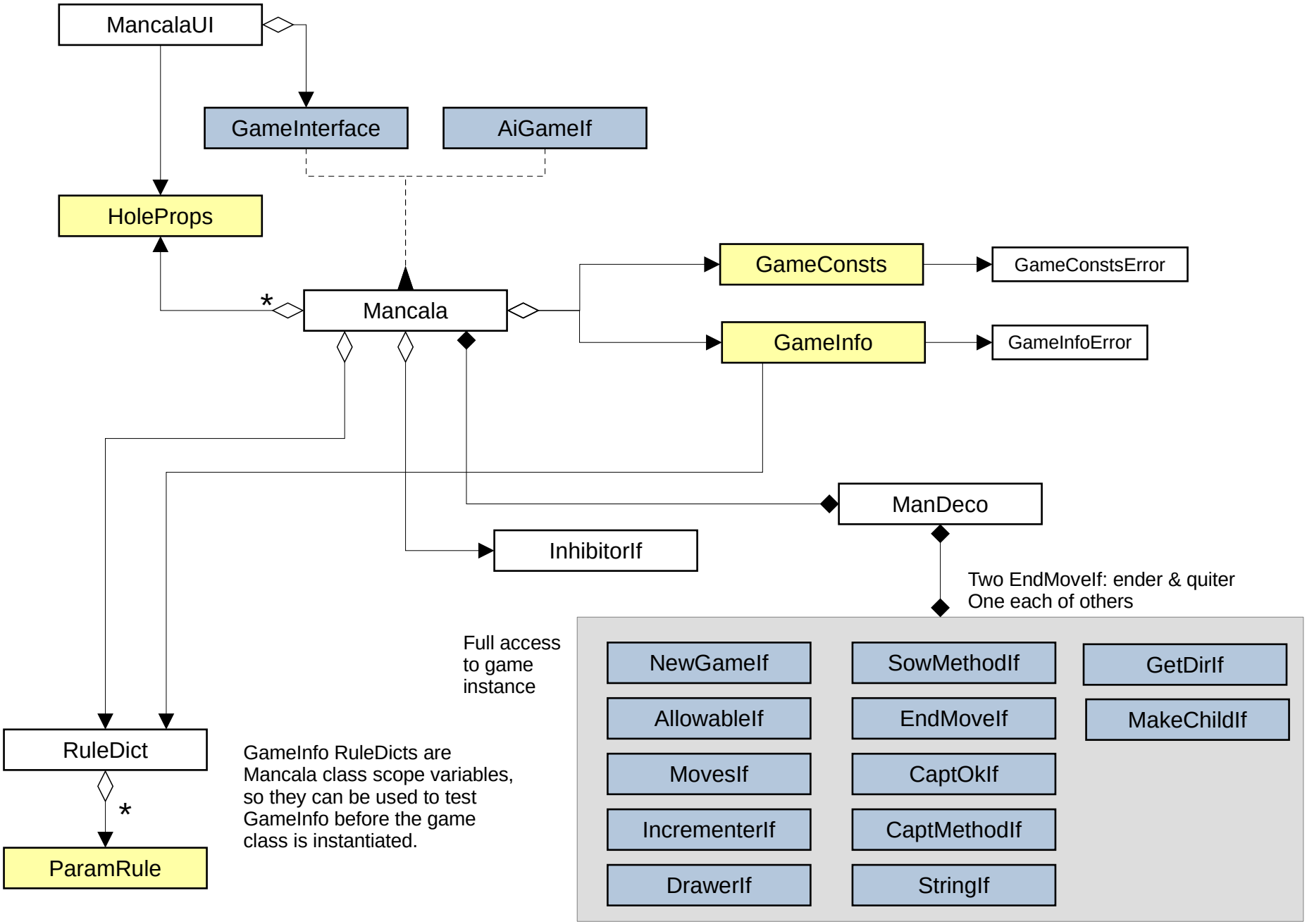
Assignments to an AniList generate SetStateVar animations.

These animator hooks are used for 5 state variable and only when they are configured for use in a game.

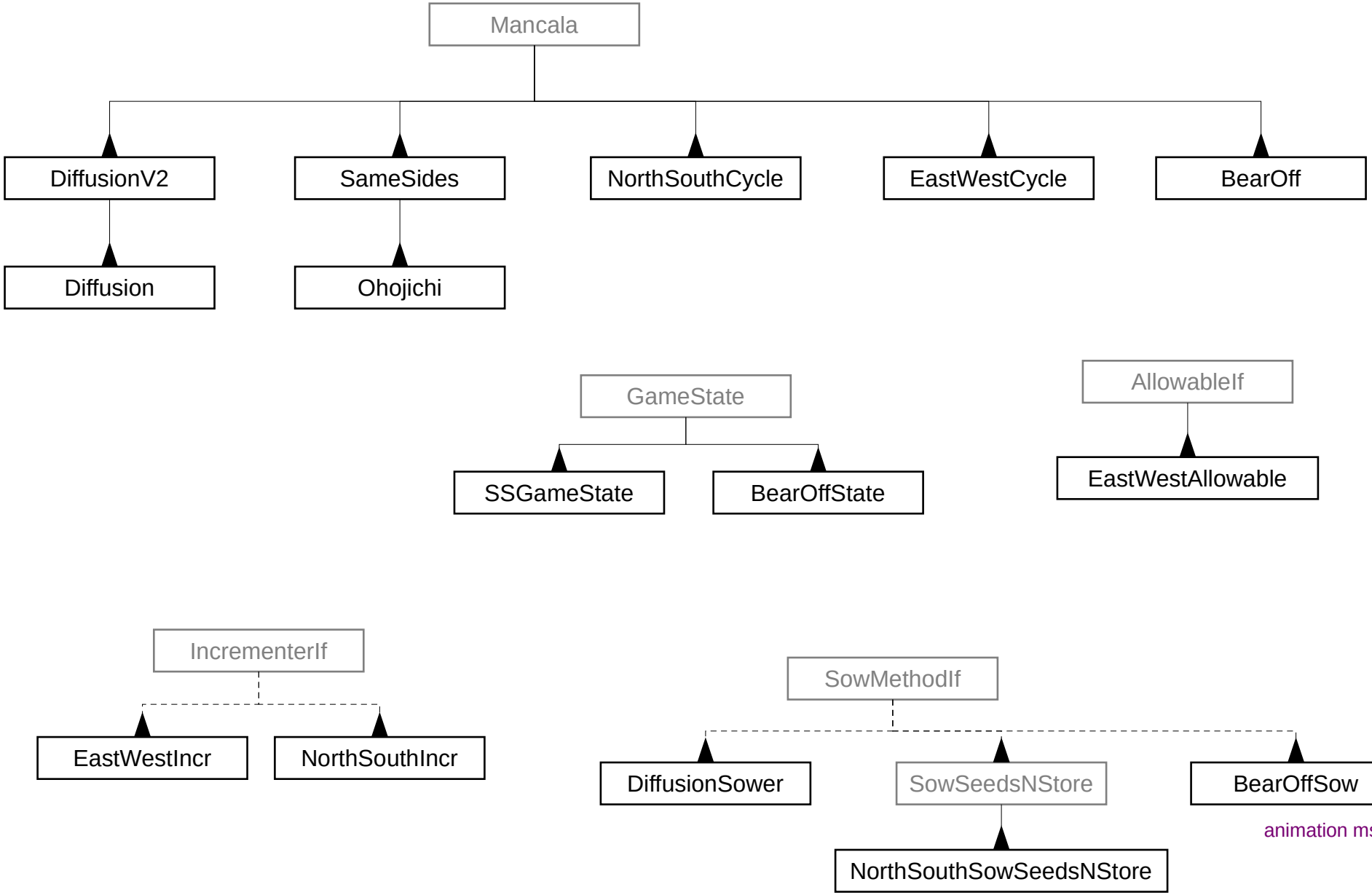
These hooks are not included if animator.ENABLED is set False.



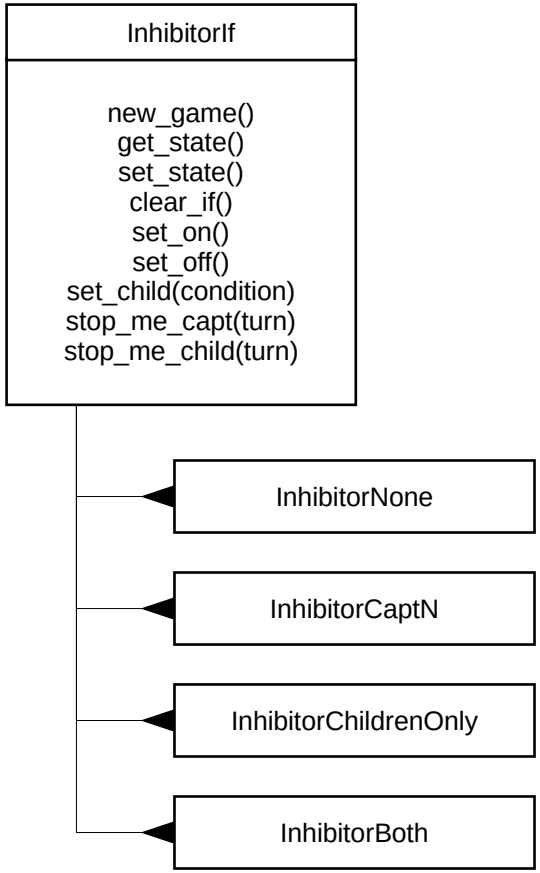
Mancala Classes



Additional Game Classes and Supporting Decorators



Import Classes for Moves



The decorator chains and button behaviors use and control the inhibitor.

MoveTpl

Moves are one of (based on game parameters):

1. position
2. (position, direction)
3. (row, position, direction)

MoveTpl prints the moves nicely.

Row is in terms of the UI, that is Top/True is 0 and Bottom/False is 1. This is the “not” of the game.turn.

Moves are created when initializing the HoleButtons for the human players and via the get_moves deco chain for the AI player.

MoveData

MoveData is used to communicate information about each move between the deco chains and individual decorators.

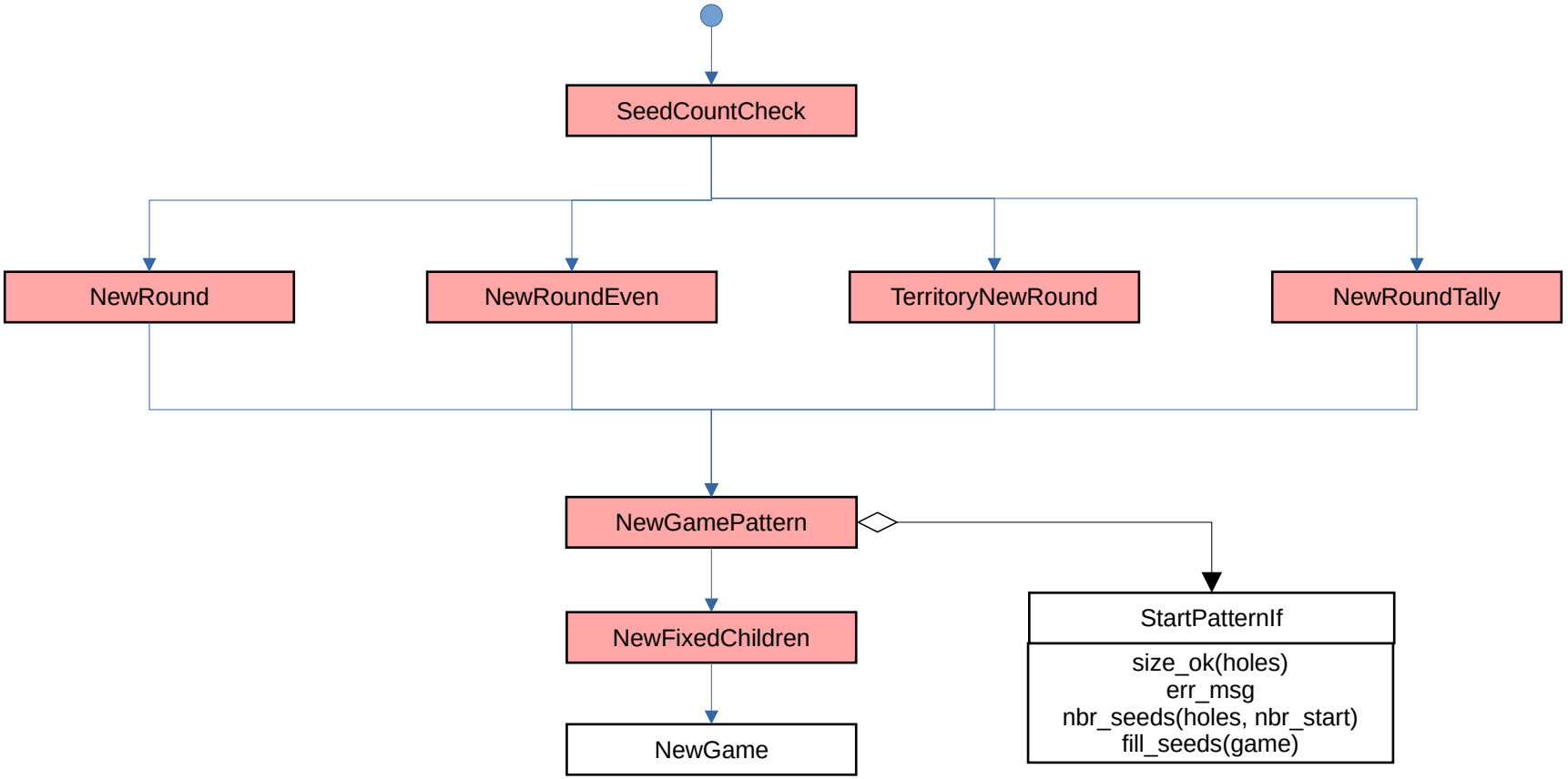
See class comment for where each field is set and/or updated.

The current move’s mdata is stored in Mancala, but anything stored directly into that could mess up the Monte Carlo Tree Search (it’s node dictionary uses a limited version of game state, which does not include Mancala.mdata).

Decorator Usage

Game Op/Step	Primary Decorator	Other Classes & Decorators Used	Description
New Game	new_game	StartPattern, inhibitor	Setups the game for initial play. Applies any prescribed moves.
Determine Drawable Holes	allow		Return a list of holes that are playable.
Collect Moves	get_moves		Return a list of possible moves.
Draw seeds to start a move	drawer		Parse the move, determine number of seeds to sow, possibly leave one seed
Determine sow direction	get_direction		Convert the move & location into an actual sowable direction: clockwise or counter-clockwise.
Sow	sower	MoveData, incr, make_child, inhibitor	Drop the seeds into the board holes.
Capture seeds	capturer & capt_ok	MoveData, incr, make_child, inhibitor	Perform any captures.
Evaluate end of game	ender	MoveData	At the end of each move determine if the game is over: game has been won, no more moves, game outcome can’t change, etc.
Logging	get_string		Creates an ASCII string for the game.
Force end of game	quitter		The game needs to end either because of endless sow or user selection. Something fair will be done.

New Game Decorators and Chain

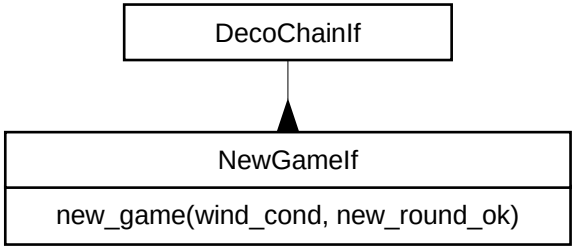


State variables changed:

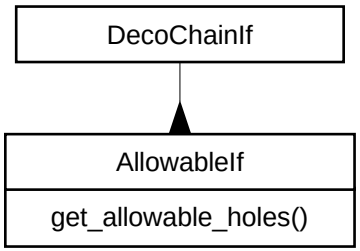
- blocked
- board
- owner
- starter
- store
- turn

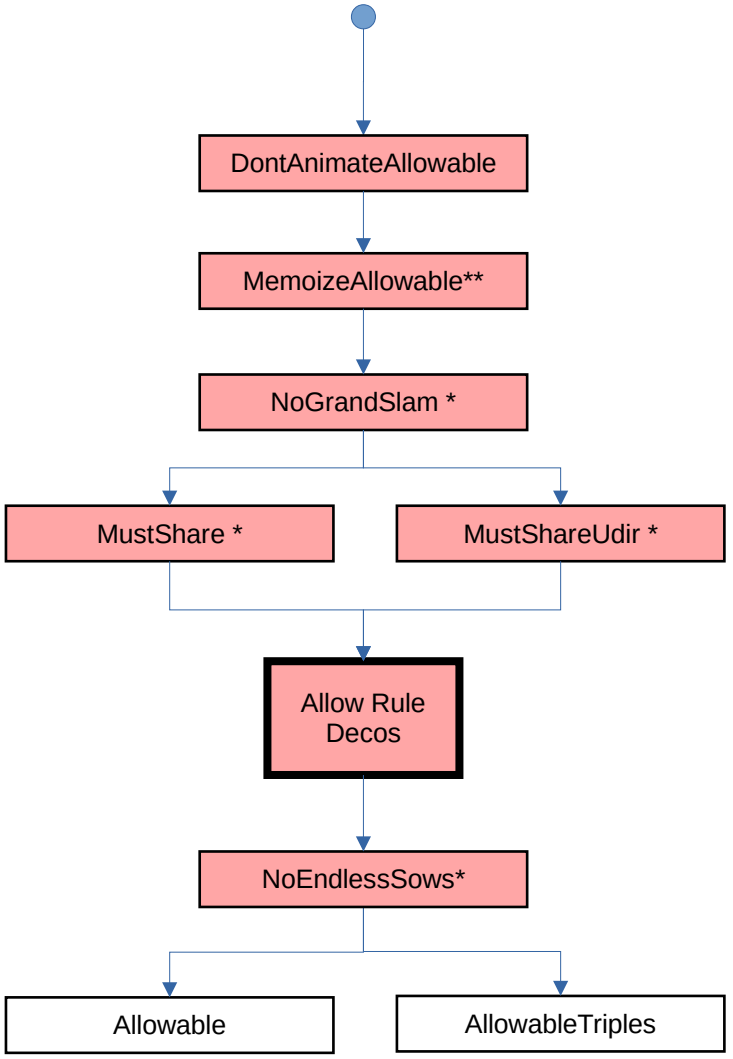
Parameters:

- blocks
- goal
- min_move
- round_starter
- round_fill
- rounds
- start_pattern



Allowables Decorators and Chain



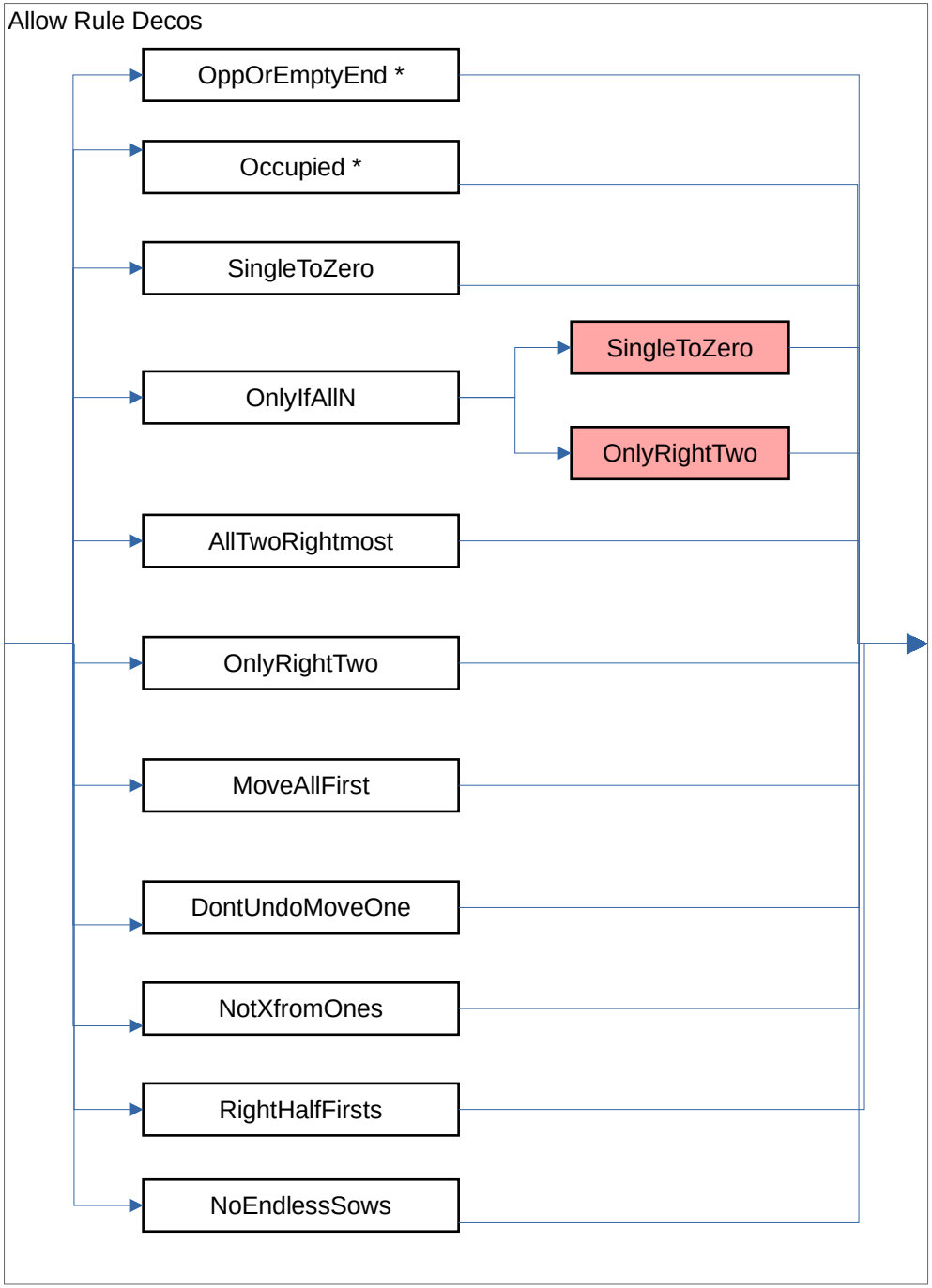


State variables read:
- turn
- board
- store
- blocked
- owner
- child
- mcount

Parameters:
- min_move
- allow_rule
- mlength
- mustshare
- grandslam
- udir_holes

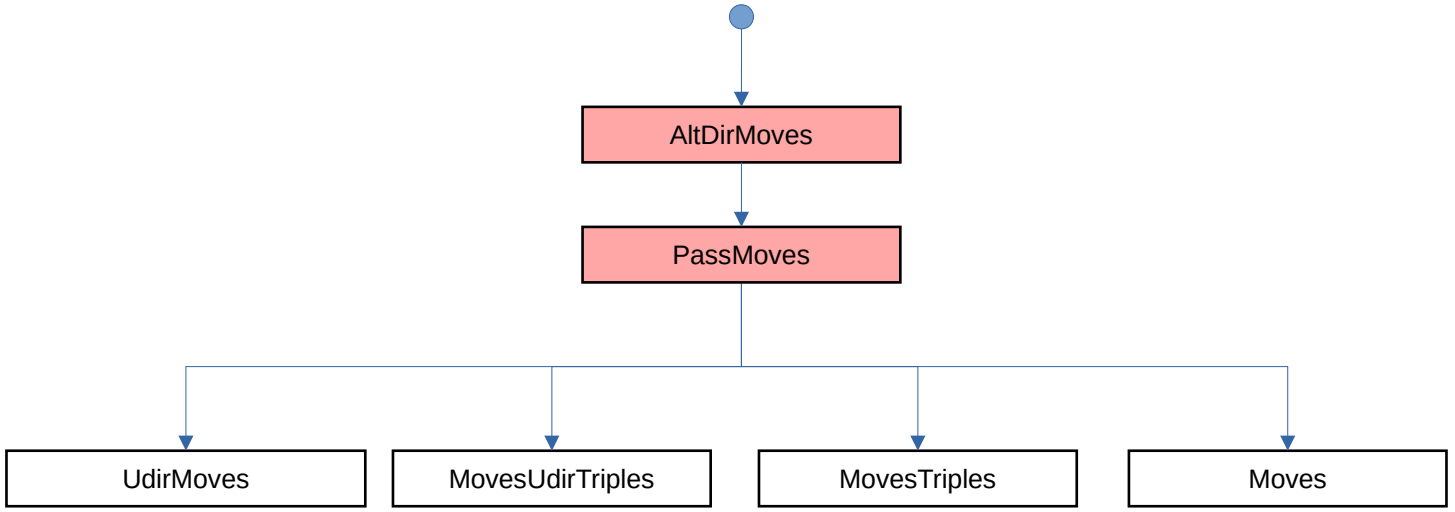
Notes:
* Simulates some portion of moves to determine allowables
** MemoizeAllowable is used for deco's that simulate moves

Allow Rule Decos



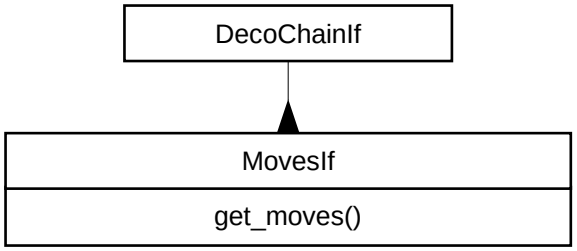
Notes:
Some allow rule decos are shown more than once for clarity.
* Simulates some portion of moves to determine allowables

Get Moves Decorators and Chain

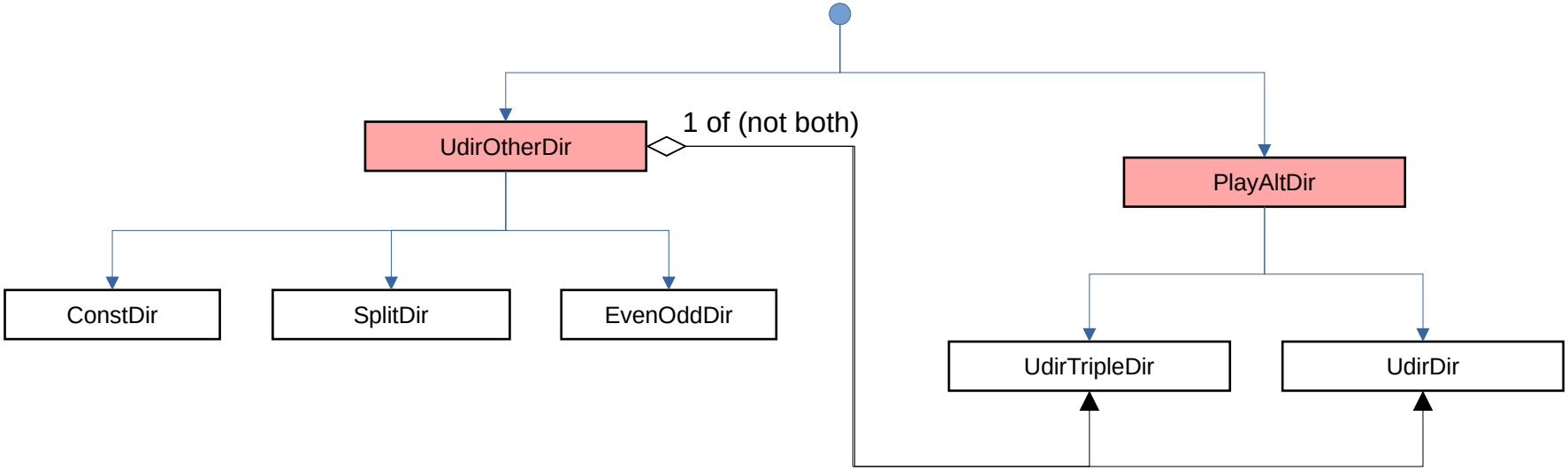


State variables read:
- blocked
- board
- owner
- starter
- store
- turn

Parameters:
- mlength
- mustpass
- sow_direct
- udir_holes
- udirect

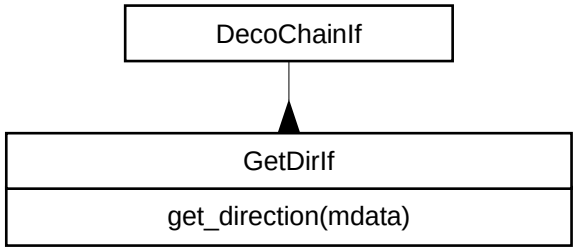


Get Direction Decorators and Chain

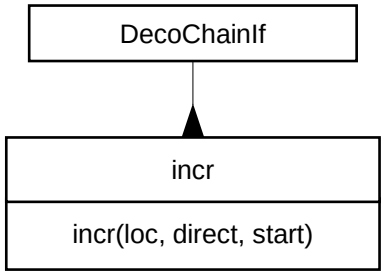


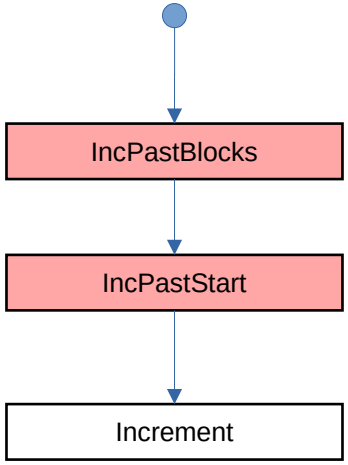
State variables read:
- mcount
- turn

Parameters:
- no_sides
- sow_direct
- udir_holes
- udirect



Incrementer Decorators and Chains





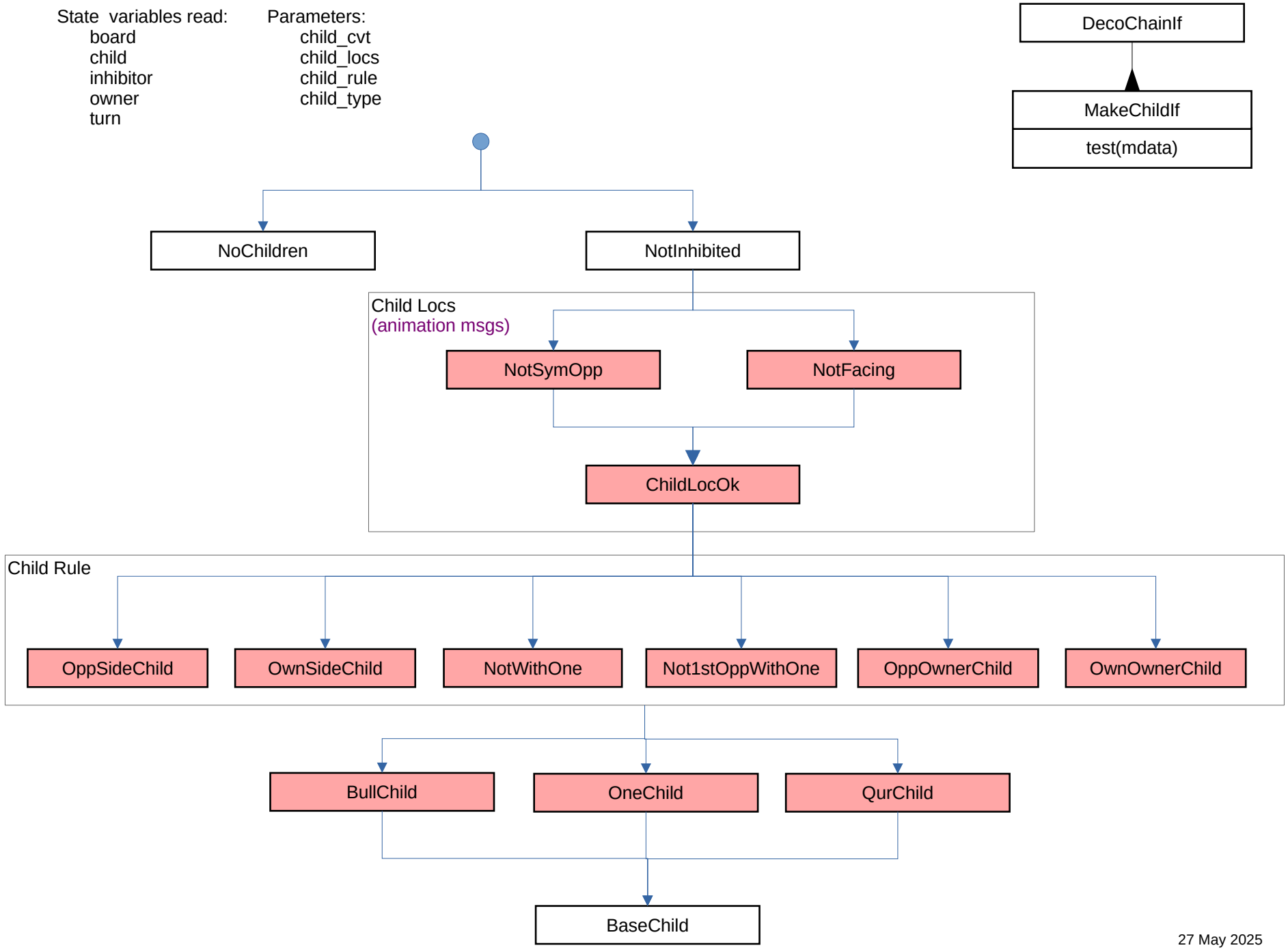
State variables read:
blocked

Parameters:
blocks
skip_start

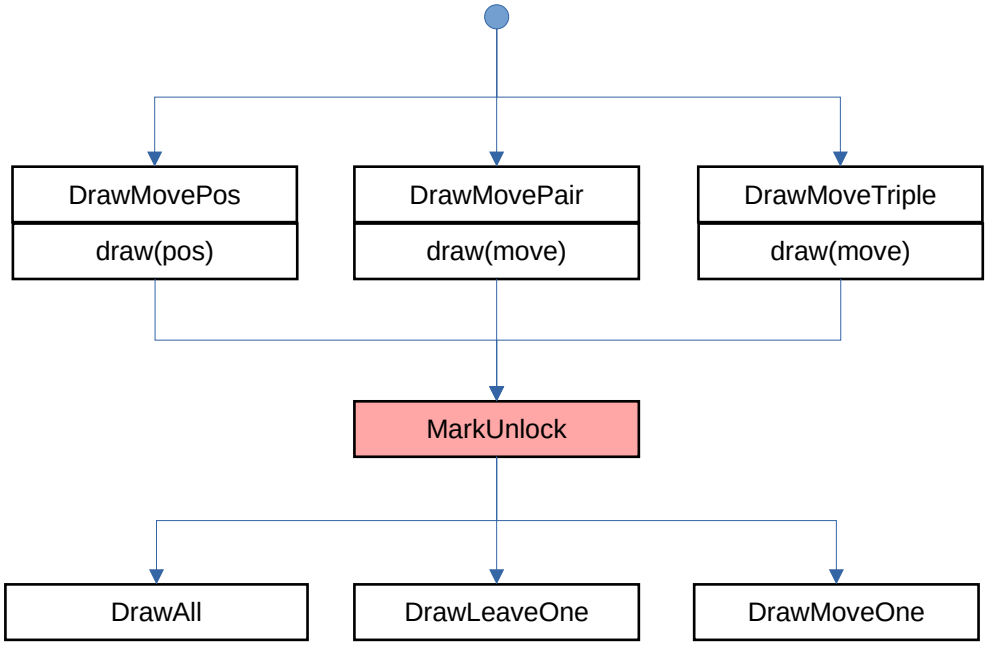
MakeChild Decorator and Chain

State variables read:
- board
- child
- inhibitor
- owner
- turn

Parameters:
- child_cvt
- child_locs
- child_rule
- child_type



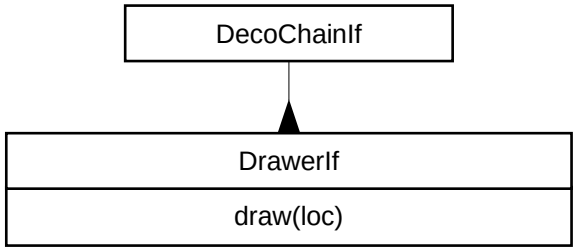
Draw Decorators and Chain



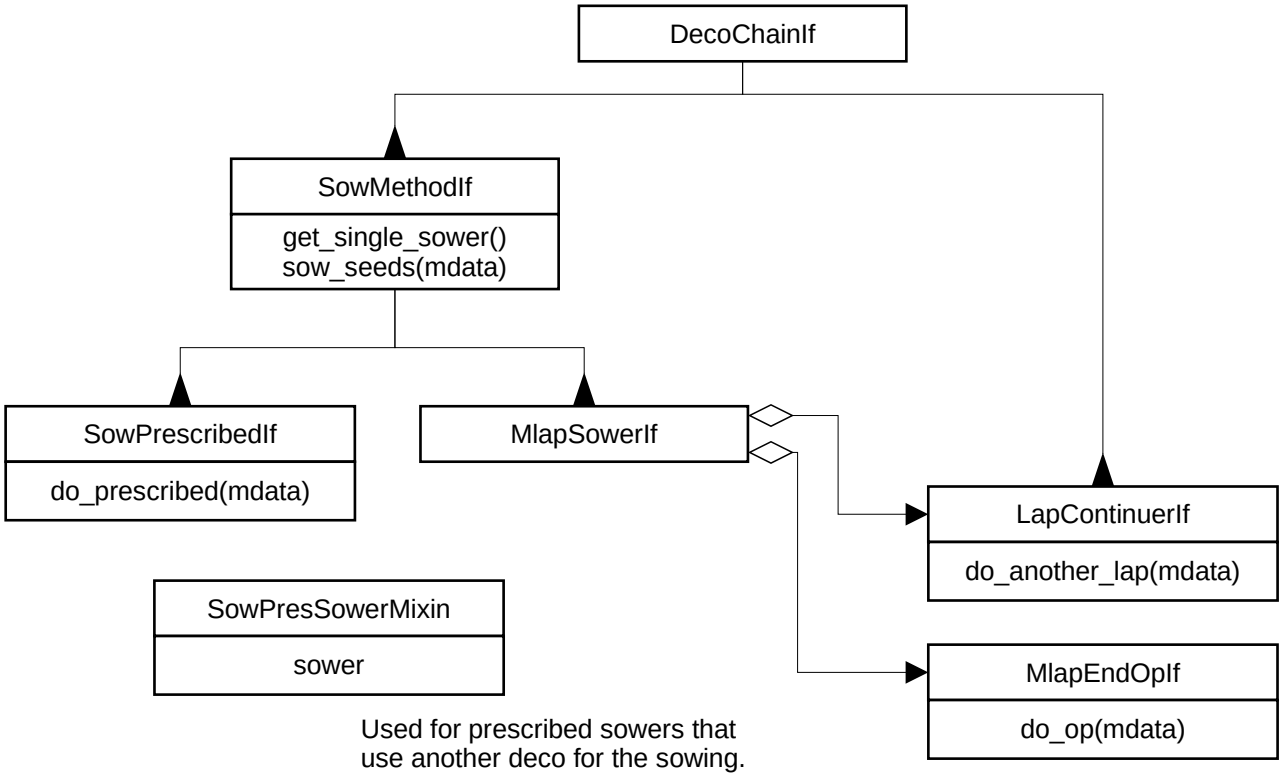
The first drawer converts the move into board location.

State variables:
- Read:
  - turn
- Changed:
  - board
  - unlocked

Parameters:
- allow_rule
- mlength
- move_one
- moveunlock
- sow_start



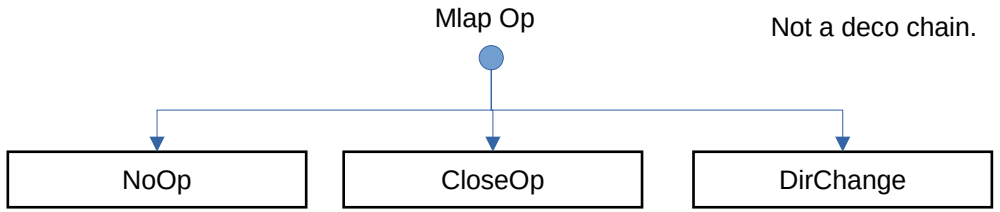
Sower Decorators



Used for prescribed sowers that use another deco for the sowing.

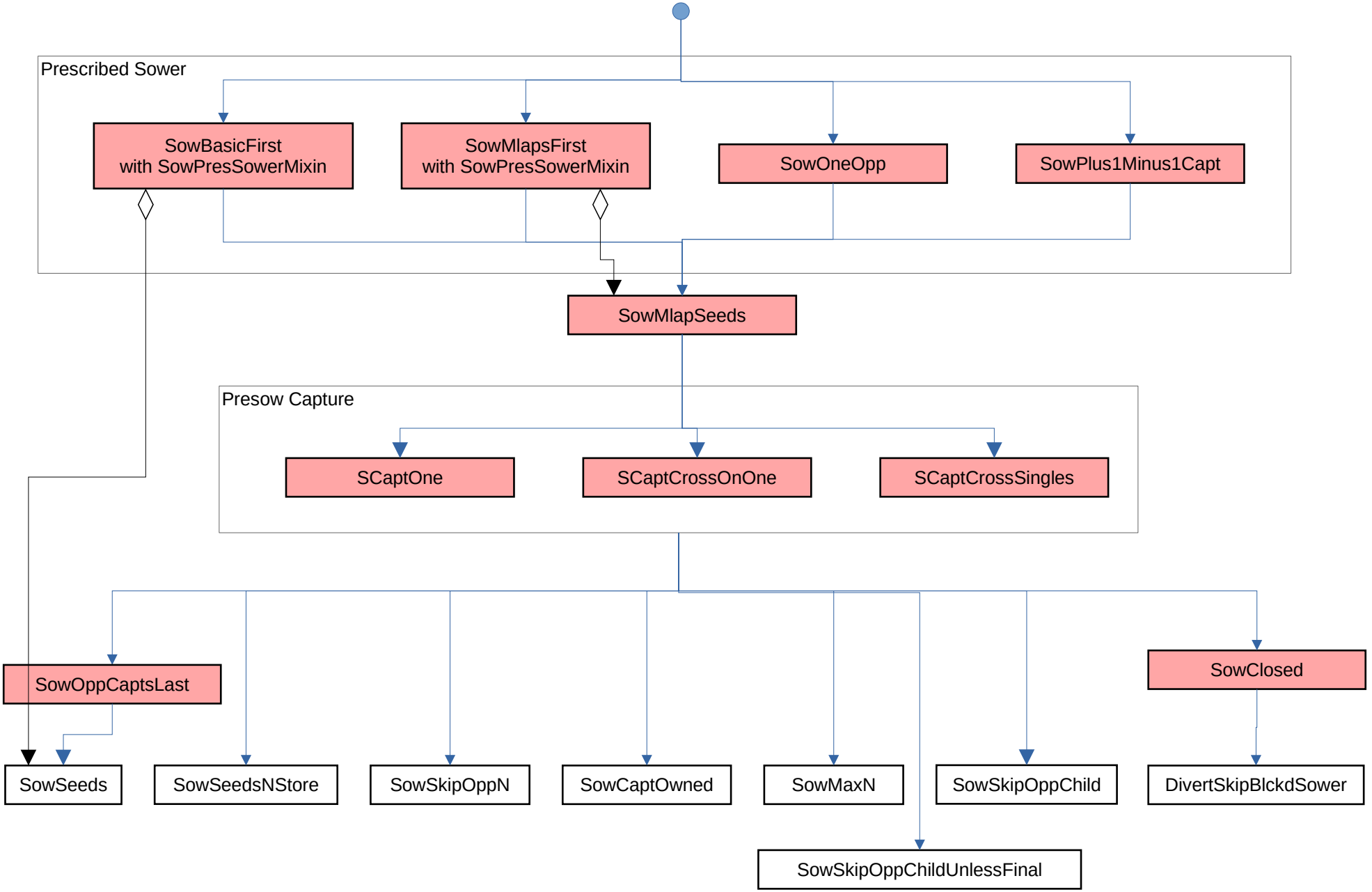
State variables:
- Reads
  - inhibitor
  - turn
  - child
  - mcount
- Changes
  - board
  - store
  - blocked

Parameters:
- capt_max
- capt_min
- capt_on
- child_type
- crosscapt
- evens
- goal
- gparam_one
- mlaps
- prescribed
- presowcapt
- sow_direct
- sow_own_store
- sow_param
- sow_rule
- visit_opp

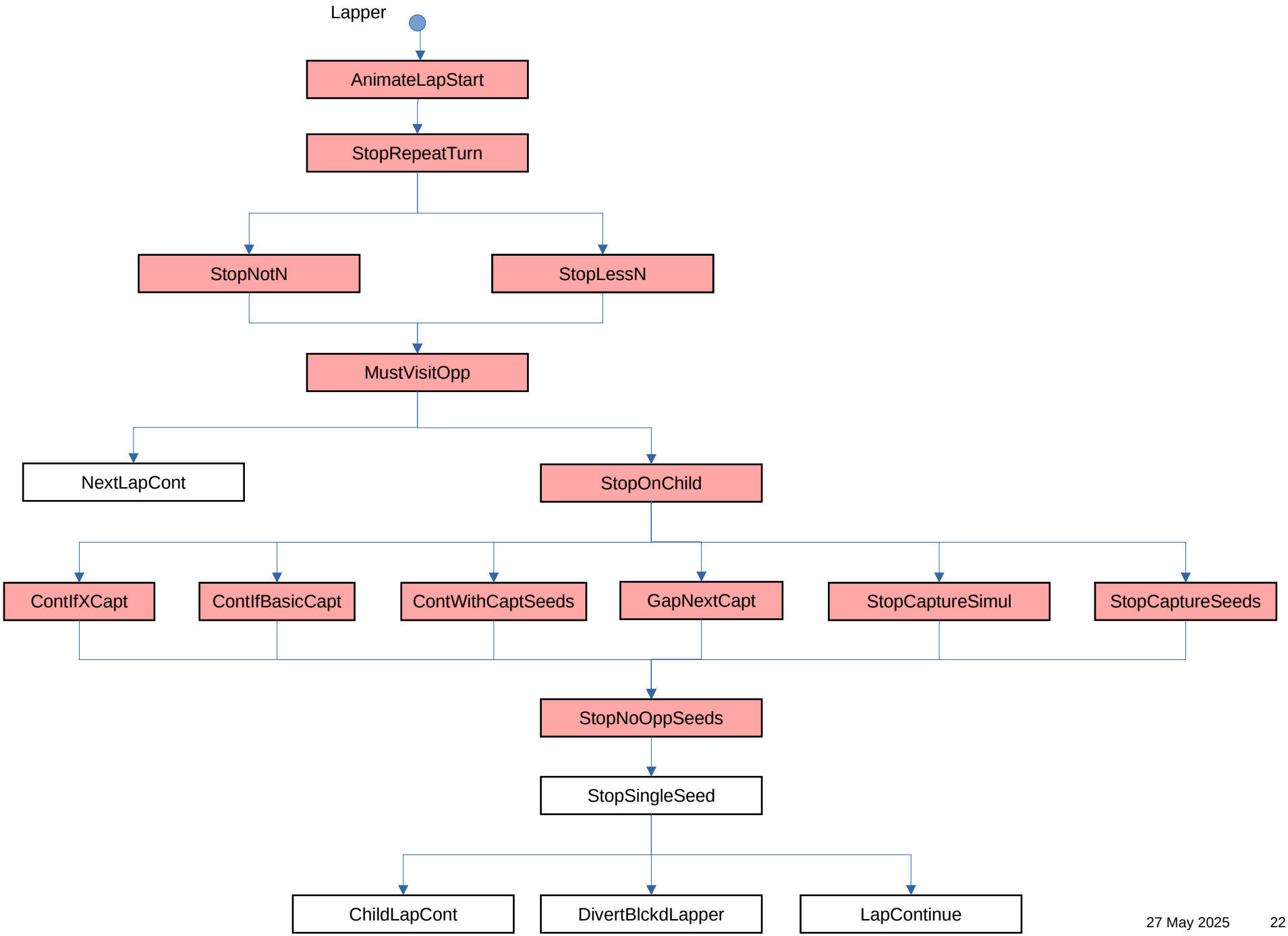


Not a deco chain.

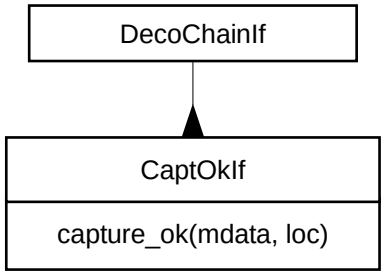
Sower Deco Chain

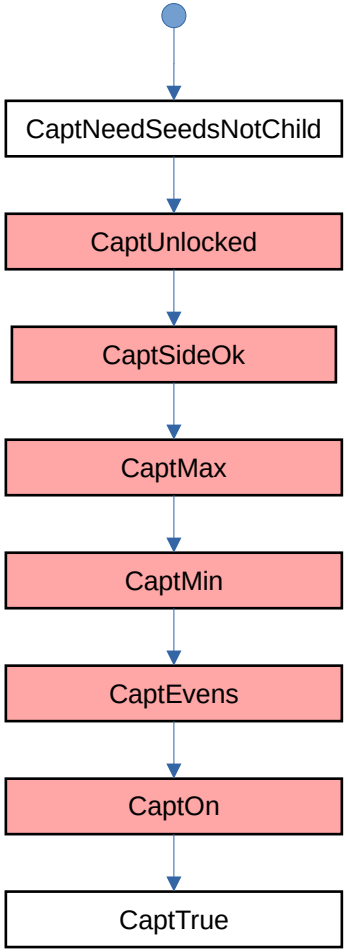


Lap Continuer Deco Chain and Mlap Operation



Capt Ok Decorators and Chains





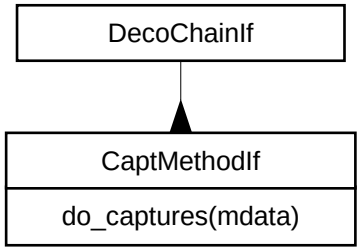
This is the Basic Capture Criteria.

These are effectively ANDed. If any deco condition is false, it returns false, otherwise it calls down the deco chain.

State variables read:
- board
- child
- turn
- unlocked

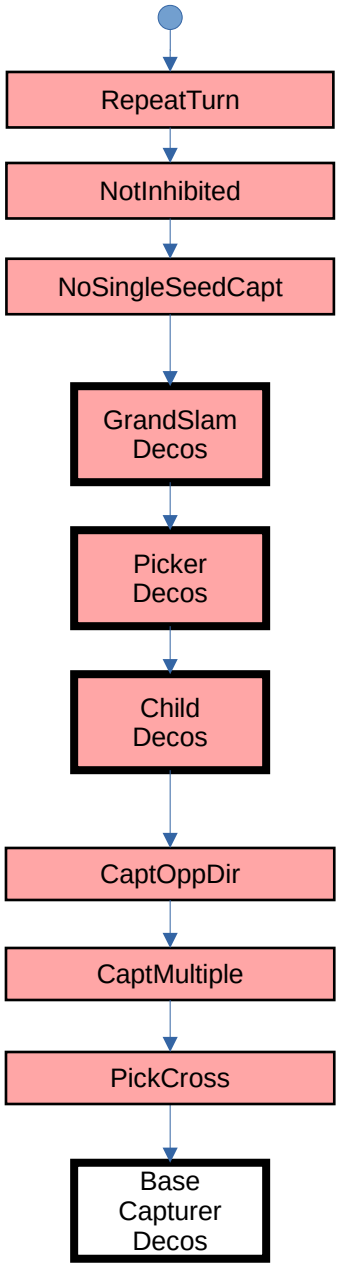
Parameters:
- capt_max
- capt_min
- capt_on
- capt_side
- moveunlock

Capturer Decorators and Chain



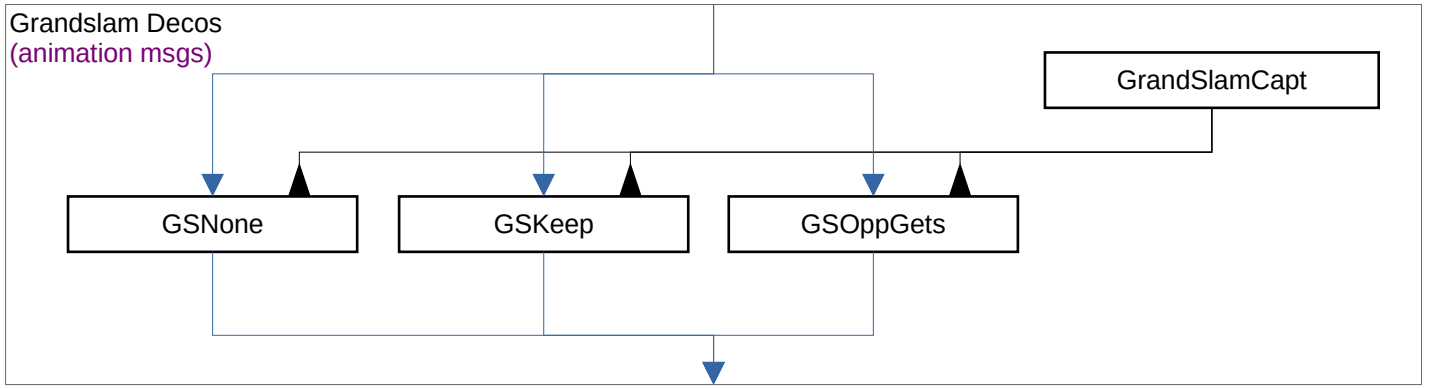
State variables
- Reads
  - inhibitor
  - starter
  - turn
- Changes
  - board
  - child
  - store

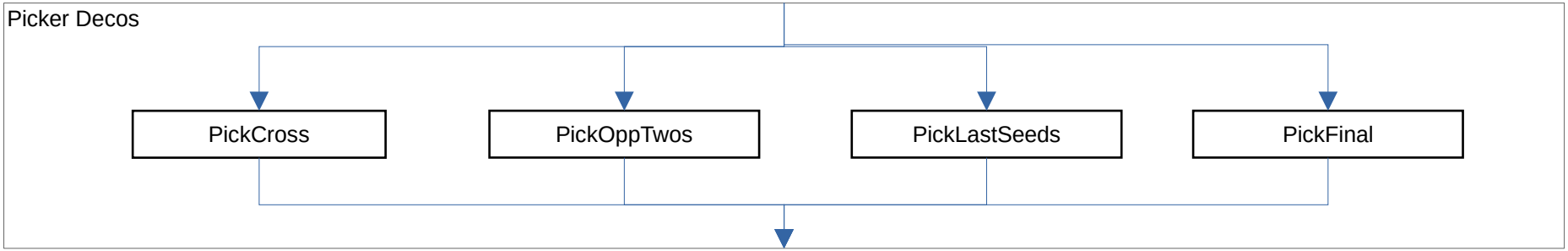
Parameters:
- capsamedir
- capt_max
- capt_min
- capt_on
- capt_rturn
- capt_side
- capt_type
- child_cvt
- child_type
- crosscapt
- evens
- grandslam
- mlaps
- multicapt
- nocaptmoves
- nosinglecapt
- pickextra
- prescribed
- round_fill
- xc_sown
- xcpickown

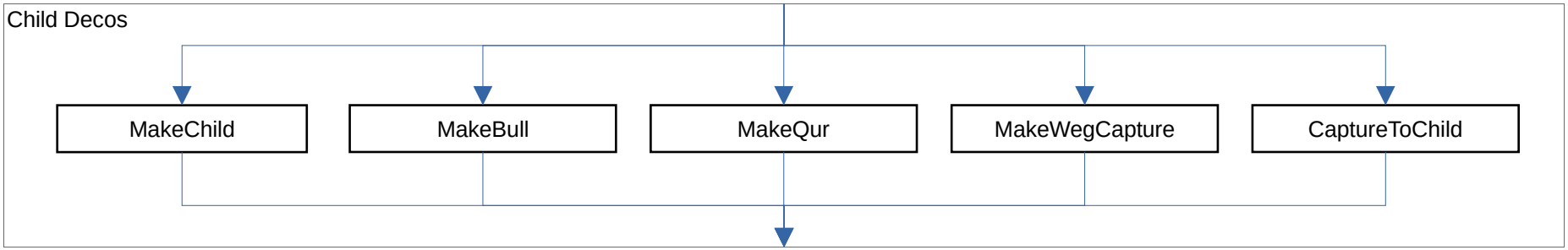


Notes:
- Not all paths are allowed: see ginfo_rules.
- Child and Grand Slam decos cannot occur together.
- Pickers do nothing when a child is made.
- PickCross is only put in the deco chain once, either in Picker Decos or after CaptMultiple.

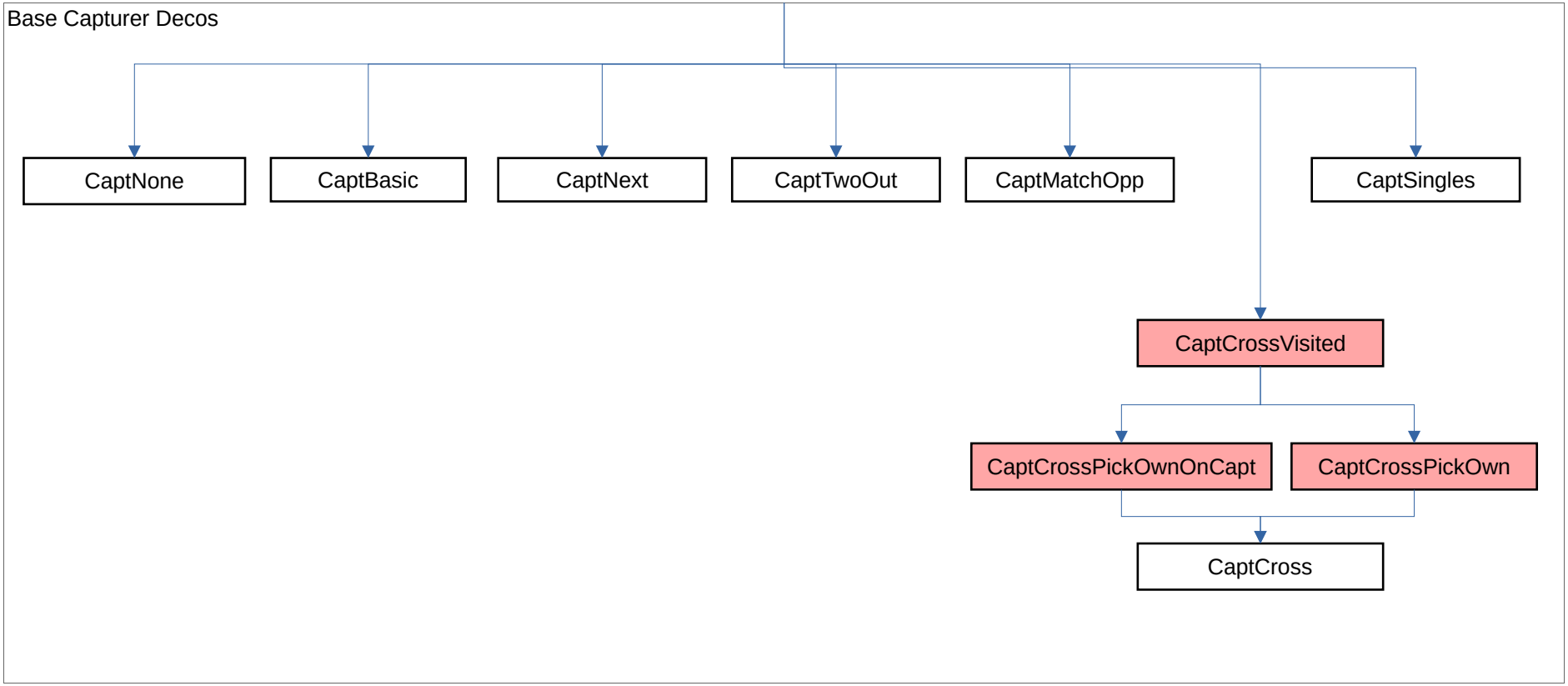
Capturer Deco Chains
(1 of 2)







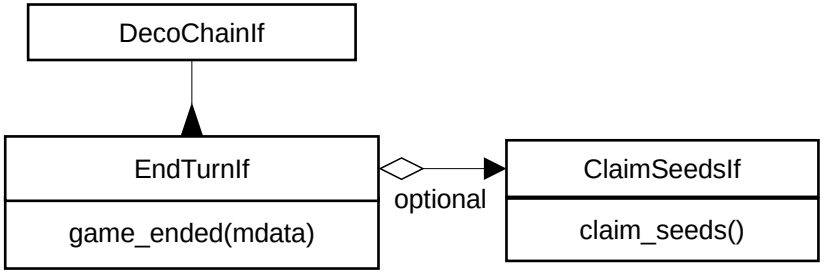
Capturer Deco Chains
(2 of 2)

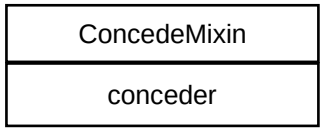


Ender & Quiter Decorators and Chains (1 of 2)

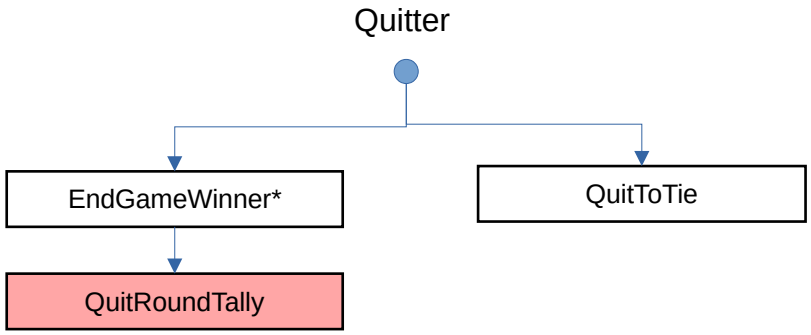
State variables:
- Reads:
  - child
  - owner
  - turn
- Changes:
  - board
  - store

Parameters:
- capt_min
- capt_next
- capt_on
- capttwoout
- child_cvt
- child_type
- crosscapt
- evens
- goal
- gparam_one
- min_move
- mlaps
- mustpass
- mustshare
- no_sides
- round_fill
- rounds
- sow_own_store
- stores
- unclaimed



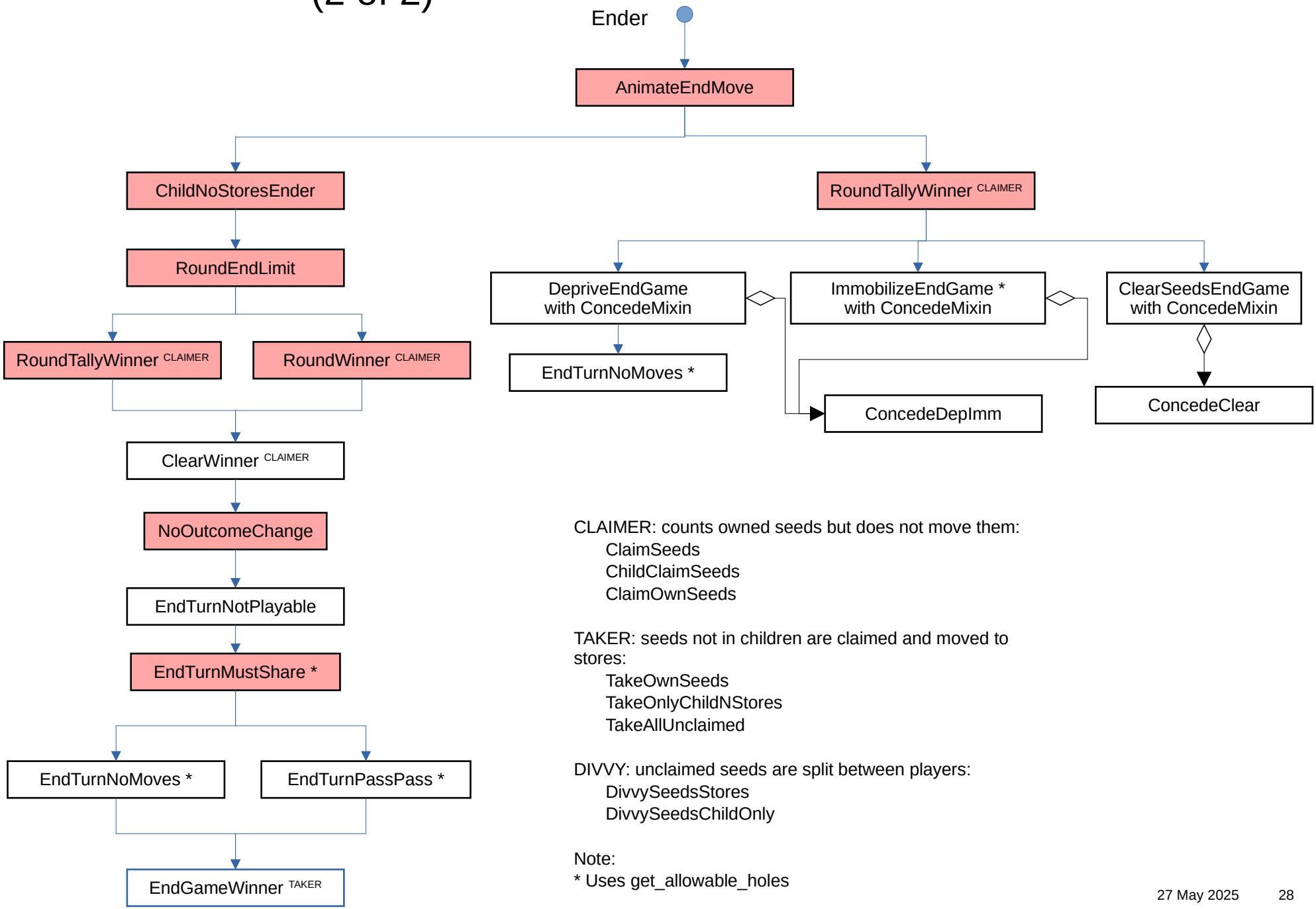


Used for enders that use a different criteria for ending when the user concedes a game.



Note:
*For EndGameWinner in the quitter: a claimer, taker or divvier is selected based on the quitter, child_type and store properties (see next page).

Ender & Quiter Decorators and Chains (2 of 2)



CLAIMER: counts owned seeds but does not move them:
- ClaimSeeds
- ChildClaimSeeds
- ClaimOwnSeeds

TAKER: seeds not in children are claimed and moved to stores:
- TakeOwnSeeds
- TakeOnlyChildNStores
- TakeAllUnclaimed

DIVVY: unclaimed seeds are split between players:
- DivvySeedsStores
- DivvySeedsChildOnly

Note:
* Uses get_allowable_holes