helper module for tinyman2 spec

About

This is helper module for the main spec of tinyman2 smart-contract.

All assertions in the `amm_approval.tl` are enumerated here with corresponding lines of code.

It greately helps to review assertions in model visualizer without searching each time through the source code.

Enumeration Assert

Consists from atoms, which names are simple concatenation of `assert` + line number in the source code.

```
module asserts

enum Assert {
-- bootstrap at amm_approval.tl:40
  assert56,
  assert59,
  assert63,
  assert72,
  assert75,
  assert81,
  assert84,
  assert93,
-- set_fee_collector at amm_approval.tl:199
  assert203,
-- set_fee_setter at amm_approval.tl:210
  assert214,
-- set_fee_manager at amm_approval.tl:221
  assert225,
-- claim_fees at amm_approval.tl:232
  assert244,
-- claim_extra at amm_approval.tl:254
  assert288,
-- set_fee at amm_approval.tl:293
  assert296,
  assert301,
  assert302,
  assert303,
  assert304,
-- amm at amm_approval.tl:312
  assert323,
-- swap at amm_approval.tl:336
  assert351,
  assert355,
  assert361,
  assert362,
  assert389,
  assert390,
  assert391,
  assert398,
  assert399,
  assert400,
-- flash_loan at amm_approval.tl:447
  assert460,
  assert462,
  assert463,
  assert466,
  assert467,
  assert468,
  assert469,
  assert471,
  assert473,
  assert474,
  assert477,
  assert481,
-- verify_flash_loan at amm_approval.tl:487
  assert499,
  assert500,
  assert501,
  assert502,
  assert504,
  assert506,
  assert507,
  assert518,
  assert528,
  assert529,
  assert530,
  assert531,
  assert532,
  assert555,
  assert562,
  assert563,
  assert564,
  assert565,
  assert568,
  assert569,
  assert570,
  assert571,
  assert572,
-- flash_swap at amm_approval.tl:597
  assert606,
  assert608,
  assert609,
  assert610,
  assert611,
  assert613,
  assert615,
  assert616,
  assert619,
  assert622,
  assert626,
-- verify_flash_swap at amm_approval.tl:640
  assert646,
  assert647,
  assert648,
  assert649,
  assert651,
  assert653,
  assert687,
-- add_liquidity at amm_approval.tl:712
  assert727,
  assert764,
  assert765,
  assert766,
  assert767,
  assert773,
  assert774,
  assert777,
  assert778,
  assert779,
  assert782,
  assert871,
  assert874,
-- add_initial_liquidity at amm_approval.tl:896
  assert909,
  assert914,
  assert915,
  assert916,
  assert917,
  assert919,
  assert922,
  assert923,
  assert926,
  assert927,
  assert928,
  assert931,
  assert932,
  assert936,
-- remove_liquidity at amm_approval.tl:949
  assert965,
  assert966,
  assert967,
  assert968,
  assert970,
  assert984,
  assert996,
  assert997,
  assert998,
  assert999,
  assert1005,
  assert1016,
  assert1034,
  assert1075,
  assert1084
}
```

Correspondence of atoms and lines from the source code

Here is a simple macro with relation between an atom name and a line from the source code.

So we can see assertions from the source code in the visualizer.

```
let assert_line {
-- bootstrap at amm_approval.tl:40
    assert56 -> "amm_approval.tl:56: assert(Txn.ApplicationArgs[0] ==
\"bootstrap\")"
  + assert59 -> "amm_approval.tl:59: assert(Txn.RekeyTo ==
Global.CurrentApplicationAddress)"
  + assert63 -> "amm_approval.tl:63: assert(asset_1_id > asset_2_id)"
  + assert72 -> "amm_approval.tl:72: assert(exists)"
  + assert75 -> "amm_approval.tl:75: assert(asset_total >=
ASSET_MIN_TOTAL)"
  + assert81 -> "amm_approval.tl:81: assert(exists)"
  + assert84 -> "amm_approval.tl:84: assert(asset_total >=
ASSET_MIN_TOTAL)"
  + assert93 -> "amm_approval.tl:93: assert(pool_address ==
sha512_256(concat(\"Program\" program)))"
-- set_fee_collector at amm_approval.tl:199
  + assert203 -> "amm_approval.tl:203: assert(user_address ==
app_global_get(\"fee_manager\"))"
-- set_fee_setter at amm_approval.tl:210
  + assert214 -> "amm_approval.tl:214: assert(user_address ==
app_global_get(\"fee_manager\"))"
-- set_fee_manager at amm_approval.tl:221
  + assert225 -> "amm_approval.tl:225: assert(user_address ==
app_global_get(\"fee_manager\"))"
-- claim_fees at amm_approval.tl:232
  + assert244 -> "amm_approval.tl:244: assert(asset_1_protocol_fees ||
asset_2_protocol_fees)"
-- claim_extra at amm_approval.tl:254
  + assert288 -> "amm_approval.tl:288: assert(asset_amount)"
-- set_fee at amm_approval.tl:293
  + assert296 -> "amm_approval.tl:296: assert(user_address ==
app_global_get(\"fee_setter\"))"
  + assert301 -> "amm_approval.tl:301: assert(total_fee_share <= 100)"
  + assert302 -> "amm_approval.tl:302: assert(total_fee_share >= 1)"
  + assert303 -> "amm_approval.tl:303: assert(protocol_fee_ratio <= 10)"
  + assert304 -> "amm_approval.tl:304: assert(protocol_fee_ratio >= 3)"
-- amm at amm_approval.tl:312
  + assert323 -> "amm_approval.tl:323: assert(app_local_get(1, \"lock\")
== (Txn.ApplicationArgs[0] == \"verify_flash_swap\"))"
-- swap at amm_approval.tl:336
  + assert351 -> "amm_approval.tl:351:
assert(Gtxn[input_txn_index].Receiver == pool_address)"
  + assert355 -> "amm_approval.tl:355:
assert(Gtxn[input_txn_index].AssetReceiver == pool_address)"
  + assert361 -> "amm_approval.tl:361:
assert(Gtxn[input_txn_index].Sender == user_address)"
  + assert362 -> "amm_approval.tl:362: assert(input_amount)"
  + assert389 -> "amm_approval.tl:389: assert(output_amount)"
  + assert390 -> "amm_approval.tl:390: assert(total_fee_amount)"
  + assert391 -> "amm_approval.tl:391: assert(output_amount >=
min_output)"
  + assert398 -> "amm_approval.tl:398: assert(output_amount)"
  + assert399 -> "amm_approval.tl:399: assert(total_fee_amount)"
  + assert400 -> "amm_approval.tl:400: assert(input_amount >=
required_input_amount)"
-- flash_loan at amm_approval.tl:447
  + assert460 -> "amm_approval.tl:460: assert(index_diff > 2)"
  + assert462 -> "amm_approval.tl:462: assert(index_diff > 1)"
  + assert463 -> "amm_approval.tl:463: assert(asset_1_amount ||
asset_2_amount)"
```

```
  + assert466 -> "amm_approval.tl:466:
assert(Gtxn[verify_flash_loan_txn_index].TypeEnum == Appl)"
  + assert467 -> "amm_approval.tl:467:
assert(Gtxn[verify_flash_loan_txn_index].OnCompletion == NoOp)"
  + assert468 -> "amm_approval.tl:468:
assert(Gtxn[verify_flash_loan_txn_index].ApplicationID ==
Global.CurrentApplicationID)"
  + assert469 -> "amm_approval.tl:469:
assert(Gtxn[verify_flash_loan_txn_index].ApplicationArgs[0] ==
\"verify_flash_loan\")"
  + assert471 -> "amm_approval.tl:471:
assert(Gtxn[verify_flash_loan_txn_index].ApplicationArgs[1] ==
Txn.ApplicationArgs[1])"
  + assert473 -> "amm_approval.tl:473:
assert(Gtxn[verify_flash_loan_txn_index].Accounts[1] == Txn.Accounts[1])"
  + assert474 -> "amm_approval.tl:474:
assert(Gtxn[verify_flash_loan_txn_index].Sender == user_address)"
  + assert477 -> "amm_approval.tl:477: assert(asset_1_amount <=
asset_1_reserves)"
  + assert481 -> "amm_approval.tl:481: assert(asset_2_amount <=
asset_2_reserves)"
-- verify_flash_loan at amm_approval.tl:487
  + assert499 -> "amm_approval.tl:499:
assert(Gtxn[flash_loan_txn_index].TypeEnum == Appl)"
  + assert500 -> "amm_approval.tl:500:
assert(Gtxn[flash_loan_txn_index].OnCompletion == NoOp)"
  + assert501 -> "amm_approval.tl:501:
assert(Gtxn[flash_loan_txn_index].ApplicationID ==
Global.CurrentApplicationID)"
  + assert502 -> "amm_approval.tl:502:
assert(Gtxn[flash_loan_txn_index].ApplicationArgs[0] == \"flash_loan\")"
  + assert504 -> "amm_approval.tl:504:
assert(Gtxn[flash_loan_txn_index].ApplicationArgs[1] ==
Txn.ApplicationArgs[1])"
  + assert506 -> "amm_approval.tl:506:
assert(Gtxn[flash_loan_txn_index].Accounts[1] == Txn.Accounts[1])"
  + assert507 -> "amm_approval.tl:507:
assert(Gtxn[flash_loan_txn_index].Sender == user_address)"
  + assert518 -> "amm_approval.tl:518: assert(asset_1_total_fee_amount)"
  + assert528 -> "amm_approval.tl:528:
assert(Gtxn[asset_1_txn_index].TypeEnum == Axfer)"
  + assert529 -> "amm_approval.tl:529:
assert(Gtxn[asset_1_txn_index].XferAsset == asset_1_id)"
  + assert530 -> "amm_approval.tl:530:
assert(Gtxn[asset_1_txn_index].AssetReceiver == pool_address)"
  + assert531 -> "amm_approval.tl:531:
assert(Gtxn[asset_1_txn_index].AssetAmount >= asset_1_repayment_amount)"
  + assert532 -> "amm_approval.tl:532:
assert(Gtxn[asset_1_txn_index].Sender == user_address)"
  + assert555 -> "amm_approval.tl:555: assert(asset_2_total_fee_amount)"
  + assert562 -> "amm_approval.tl:562:
assert(Gtxn[asset_2_txn_index].TypeEnum == Pay)"
  + assert563 -> "amm_approval.tl:563:
assert(Gtxn[asset_2_txn_index].Receiver == pool_address)"
  + assert564 -> "amm_approval.tl:564:
assert(Gtxn[asset_2_txn_index].Amount >= asset_2_repayment_amount)"
  + assert565 -> "amm_approval.tl:565:
assert(Gtxn[asset_2_txn_index].Sender == user_address)"
  + assert568 -> "amm_approval.tl:568:
assert(Gtxn[asset_2_txn_index].TypeEnum == Axfer)"
```

```
  + assert569 -> "amm_approval.tl:569:
assert(Gtxn[asset_2_txn_index].XferAsset == asset_2_id)"
  + assert570 -> "amm_approval.tl:570:
assert(Gtxn[asset_2_txn_index].AssetReceiver == pool_address)"
  + assert571 -> "amm_approval.tl:571:
assert(Gtxn[asset_2_txn_index].AssetAmount >= asset_2_repayment_amount)"
  + assert572 -> "amm_approval.tl:572:
assert(Gtxn[asset_2_txn_index].Sender == user_address)"
-- flash_swap at amm_approval.tl:597
  + assert606 -> "amm_approval.tl:606: assert(index_diff > 1)"
  + assert608 -> "amm_approval.tl:608:
assert(Gtxn[verify_flash_swap_txn_index].TypeEnum == Appl)"
  + assert609 -> "amm_approval.tl:609:
assert(Gtxn[verify_flash_swap_txn_index].OnCompletion == NoOp)"
  + assert610 -> "amm_approval.tl:610:
assert(Gtxn[verify_flash_swap_txn_index].ApplicationID ==
Global.CurrentApplicationID)"
  + assert611 -> "amm_approval.tl:611:
assert(Gtxn[verify_flash_swap_txn_index].ApplicationArgs[0] ==
\"verify_flash_swap\")"
  + assert613 -> "amm_approval.tl:613:
assert(Gtxn[verify_flash_swap_txn_index].ApplicationArgs[1] ==
Txn.ApplicationArgs[1])"
  + assert615 -> "amm_approval.tl:615:
assert(Gtxn[verify_flash_swap_txn_index].Accounts[1] == Txn.Accounts[1])"
  + assert616 -> "amm_approval.tl:616:
assert(Gtxn[verify_flash_swap_txn_index].Sender == user_address)"
  + assert619 -> "amm_approval.tl:619: assert(asset_1_output_amount ||
asset_2_output_amount)"
  + assert622 -> "amm_approval.tl:622: assert(asset_1_output_amount <=
asset_1_reserves)"
  + assert626 -> "amm_approval.tl:626: assert(asset_2_output_amount <=
asset_2_reserves)"
-- verify_flash_swap at amm_approval.tl:640
  + assert646 -> "amm_approval.tl:646:
assert(Gtxn[flash_swap_txn_index].TypeEnum == Appl)"
  + assert647 -> "amm_approval.tl:647:
assert(Gtxn[flash_swap_txn_index].OnCompletion == NoOp)"
  + assert648 -> "amm_approval.tl:648:
assert(Gtxn[flash_swap_txn_index].ApplicationID ==
Global.CurrentApplicationID)"
  + assert649 -> "amm_approval.tl:649:
assert(Gtxn[flash_swap_txn_index].ApplicationArgs[0] == \"flash_swap\")"
  + assert651 -> "amm_approval.tl:651:
assert(Gtxn[flash_swap_txn_index].ApplicationArgs[1] ==
Txn.ApplicationArgs[1])"
  + assert653 -> "amm_approval.tl:653:
assert(Gtxn[flash_swap_txn_index].Accounts[1] == Txn.Accounts[1])"
  + assert687 -> "amm_approval.tl:687: assert(asset_1_total_fee_amount ||
asset_2_total_fee_amount)"
-- add_liquidity at amm_approval.tl:712
  + assert727 -> "amm_approval.tl:727: assert(issued_pool_tokens)"
  + assert764 -> "amm_approval.tl:764:
assert(Gtxn[asset_1_txn_index].TypeEnum == Axfer)"
  + assert765 -> "amm_approval.tl:765:
assert(Gtxn[asset_1_txn_index].AssetReceiver == pool_address)"
  + assert766 -> "amm_approval.tl:766:
assert(Gtxn[asset_1_txn_index].XferAsset == asset_1_id)"
  + assert767 -> "amm_approval.tl:767:
assert(Gtxn[asset_1_txn_index].Sender == user_address)"
```

```
  + assert773 -> "amm_approval.tl:773:
assert(Gtxn[asset_2_txn_index].TypeEnum == Pay)"
  + assert774 -> "amm_approval.tl:774:
assert(Gtxn[asset_2_txn_index].Receiver == pool_address)"
  + assert777 -> "amm_approval.tl:777:
assert(Gtxn[asset_2_txn_index].TypeEnum == Axfer)"
  + assert778 -> "amm_approval.tl:778:
assert(Gtxn[asset_2_txn_index].AssetReceiver == pool_address)"
  + assert779 -> "amm_approval.tl:779:
assert(Gtxn[asset_2_txn_index].XferAsset == asset_2_id)"
  + assert782 -> "amm_approval.tl:782:
assert(Gtxn[asset_2_txn_index].Sender == user_address)"
  + assert871 -> "amm_approval.tl:871: assert(pool_tokens_out)"
  + assert874 -> "amm_approval.tl:874: assert(pool_tokens_out >=
min_output)"
-- add_initial_liquidity at amm_approval.tl:896
  + assert909 -> "amm_approval.tl:909: assert(issued_pool_tokens == 0)"
  + assert914 -> "amm_approval.tl:914:
assert(Gtxn[asset_1_txn_index].TypeEnum == Axfer)"
  + assert915 -> "amm_approval.tl:915:
assert(Gtxn[asset_1_txn_index].AssetReceiver == pool_address)"
  + assert916 -> "amm_approval.tl:916:
assert(Gtxn[asset_1_txn_index].XferAsset == asset_1_id)"
  + assert917 -> "amm_approval.tl:917:
assert(Gtxn[asset_1_txn_index].Sender == user_address)"
  + assert919 -> "amm_approval.tl:919: assert(asset_1_amount)"
  + assert922 -> "amm_approval.tl:922:
assert(Gtxn[asset_2_txn_index].TypeEnum == Pay)"
  + assert923 -> "amm_approval.tl:923:
assert(Gtxn[asset_2_txn_index].Receiver == pool_address)"
  + assert926 -> "amm_approval.tl:926:
assert(Gtxn[asset_2_txn_index].TypeEnum == Axfer)"
  + assert927 -> "amm_approval.tl:927:
assert(Gtxn[asset_2_txn_index].AssetReceiver == pool_address)"
  + assert928 -> "amm_approval.tl:928:
assert(Gtxn[asset_2_txn_index].XferAsset == asset_2_id)"
  + assert931 -> "amm_approval.tl:931: assert(asset_2_amount)"
  + assert932 -> "amm_approval.tl:932:
assert(Gtxn[asset_2_txn_index].Sender == user_address)"
  + assert936 -> "amm_approval.tl:936: assert(issued_pool_tokens >
LOCKED_POOL_TOKENS)"
-- remove_liquidity at amm_approval.tl:949
  + assert965 -> "amm_approval.tl:965:
assert(Gtxn[pool_token_txn_index].TypeEnum == Axfer)"
  + assert966 -> "amm_approval.tl:966:
assert(Gtxn[pool_token_txn_index].AssetReceiver == pool_address)"
  + assert967 -> "amm_approval.tl:967:
assert(Gtxn[pool_token_txn_index].XferAsset == pool_token_asset_id)"
  + assert968 -> "amm_approval.tl:968:
assert(Gtxn[pool_token_txn_index].Sender == user_address)"
  + assert970 -> "amm_approval.tl:970: assert(removed_pool_token_amount)"
  + assert984 -> "amm_approval.tl:984: assert(asset_1_amount &&
asset_2_amount)"
  + assert996 -> "amm_approval.tl:996: assert(Txn.Assets[0] ==
asset_1_id)"
  + assert997 -> "amm_approval.tl:997: assert(Txn.Assets[1] ==
asset_2_id)"
  + assert998 -> "amm_approval.tl:998: assert(asset_1_amount >=
min_output_1)"
  + assert999 -> "amm_approval.tl:999: assert(asset_2_amount >=
min_output_2)"
  + assert1005 -> "amm_approval.tl:1005: assert(issued_pool_tokens > 0)"
  + assert1016 -> "amm_approval.tl:1016: assert(final_output_amount >=
min_output_1)"
  + assert1034 -> "amm_approval.tl:1034: assert(final_output_amount >=
min_output_2)"
  + assert1075 -> "amm_approval.tl:1075: assert((itob(app_local_get(1,
\"asset_1_reserves\")) b* itob(app_local_get(1, \"asset_2_reserves\")))
b<= (itob(asset_1_reserves - asset_1_poolers_fee_amount) b*
itob(asset_2_reserves - asset_2_poolers_fee_amount)))"
  + assert1084 -> "amm_approval.tl:1084: assert(tmp_initial b<=
tmp_final)"
}
```