**All of these Error and Warning messages are visible in the C++ source code of the Solidity compiler, available at** <https://github.com/ethereum/solidity/tree/develop/libsolidity/analysis>

**Folders done :**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Legend** | Not listed | listed | under dev | Finished ☺ | Included in website | Nothing |

|  |  |  |
| --- | --- | --- |
| **Status** | **Folder** | **Description** |
| listed | /libdevcore |  |
| listed | /libevmasm |  |
| listed | /liblangutil | **Very interesting folder** |
| Not listed | /liblll | **Not sure since apparently, it is going to be deprecated** <https://solidity.readthedocs.io/en/v0.5.11/lll.html> |
| Nothing | /libsolc | *just* **file not found *on the libsolc.cpp file, line 56*** |
| under dev | /libsolidity |  |
| Finished ☺ | /analysis | 67 errors not listed, not too bad |
| Finished ☺ | /ast | I actually have one file left to do, but it’s not urgent. |
| Finished ☺ | /codegen | Finished ☺ |
| Not listed | /formal | This can wait until the end of the project. |
| **listed** | /interface |  |
| Finished ☺ | /parsing | 7 left, double check changes in new Solidity compiler |
| Not listed | /libyul  **Coming soon** | *For later, not really documented but really interesting ☺, would give me some edges, and specialty as well ), see* <https://solidity.readthedocs.io/en/v0.5.11/yul.html> |
| listed | ./ |  |
| Not listed | /backends |  |
| listed | /evm |  |
| Not listed | /wasm |  |
| Not listed | /optimiser | ***This is a huuuuuggggeeee folder !*** |
| Not listed | /lllc | LLLC, the Lovely Little Language Compiler  <https://solidity.readthedocs.io/en/v0.5.11/yul.html>  **Not sure, since it might be deprecated** |
| Not listed | /scripts |  |
| Not listed | ./ |  |
| Not listed | /bytecodecompare | Just a Python file of interest |
| Nothing | /deps-ppa | *Only shell files* |
| Nothing | /travis-emscripten | *Only shell files* |
| Nothing | /snap | Just a .yaml file |
| Not listed | /solc | Include all the details of the CLI tool (options flags, outputs, etc…)  **Really interesting read, and only 3 files ☺**  **To look afterwards** |
| under dev | /test | Just /parsing and /import folder left |

**Good article to understand the compiler >** <https://media.consensys.net/installing-ethereum-compilers-61d701e78f6>

**Folder :** /libsolidity/analysis

* *constantEvaluator.cpp*
* *contractLevelChecker.cpp*
* *ControlFlowAnalyzer.cpp*
* *ControlFlowBuilder.cpp (Nothing)*
* *ControlFlowGraph.cpp (Nothing)*
* *DeclarationContainer.cpp*
* *DocStringAnalyser.cpp*
* *GlobalContext.cpp (Nothing)*
* *NameAndTypeResolver.cpp*
* *PostTypeChecker.cpp*
* *ReferencesResolver.cpp*
* *StaticAnalyzer.cpp*
* *SyntaxChecker.cpp*
* *TypeChecker.cpp*
* *ViewPureChecker.cpp*
* **ConstantEvaluator.cpp**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Error Type** | **Message** | **Source File** | **Description** |  |
| fatalTypeError | "Operator " + string(TokenTraits::toString(\_operation.getOperator())) + " not compatible with types " + left->toString() + " and " + right->toString() | [Line 48-56](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ConstantEvaluator.cpp#L48-L56) |  |  |
| fatalTypeError | Cyclic constant definition (or maximum recursion depth exhausted). | [Line 84-85](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ConstantEvaluator.cpp#L84-L85) | Happen if we go over 32 bytes. (for m\_depth, I am not sure what it is tho…) |  |

* **ContractLevelChecker.cpp**

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Type** | **Message** | **Source File** | **Example** |
| declarationError | "Another declaration is here:", constructor->location()), "More than one constructor defined." |  | constructor() public {  owner = msg.sender;  }    constructor() public {  map1[msg.sender] = 40;  } |
| declarationError | "Another declaration is here:", fallback->location()), "Only one fallback function is allowed." |  |  |
| findDuplicateDefinitions | Function with same name and arguments defined twice. |  | function foo() public pure returns (string memory) {  return "foo";  }    function foo() public pure returns (string memory) {  return "bar";  } |
| findDuplicateDefinitions | Event with same name and arguments defined twice. |  | event Deposit(address \_address, uint \_deposit);  event Deposit(address \_address, uint \_deposit); |
|  | Other declaration is here:", overloads[j]->location()  **This applies to anything in general (functions, events…)** | [Line 118](https://github.com/ethereum/solidity/blob/efd8d8fe5eced023476af71491e9eae3dbde4d87/libsolidity/analysis/ContractLevelChecker.cpp#L118) | function foo() public pure returns (string memory) {  return "foo";  }    function foo() public pure returns (string memory) {  return "bar";  } |
| typeError | Override changes modifier signature. |  |  |
| typeError | Override changes modifier to function. |  |  |
| typeError / overrideError | Overriding function return types differ. |  | interface Greetings {  function hello() external pure returns (string memory);  function goodbye() external pure returns (string memory);  }  contract FrenchContract is Greetings {    function hello() external pure returns (string memory) {  return "Bonjour";  }  }  contract RobotContract is Greetings {    function hello() external pure returns (uint) {  return 1;  }    } |
| typeError / overrideError | Overriding function visibility differs. |  |  |
| typeError / overrideError | "Overriding function changes state mutability from \"" + stateMutabilityToString(\_super.stateMutability()) + "\" to \"" + stateMutabilityToString(\_function.stateMutability()) + "\"." |  |  |
| typeError | Overridden function is here:", super.location() |  |  |
| typeError | Redeclaring an already implemented function as abstract |  |  |
| declarationError | Modifier-style base constructor call without arguments. |  |  |
|  | Second constructor call is here:", \_argumentNode->location() |  |  |
|  | First constructor call is here: ", \_argumentNode->location() |  |  |
|  | Second constructor call is here: ", previousNode->location() |  |  |
|  | Base constructor arguments given twice. |  |  |
| typeError | Non-empty \"returns\" directive for constructor. |  |  |
| typeError | "Constructor must be payable or non-payable, but is \"" + stateMutabilityToString(constructor->stateMutability()) + "\"." |  | constructor(bytes32 \_name) public **view** {} |
| typeError | Constructor must be public or internal. |  | constructor(bytes32 \_name) **private** {} |
| typeError | Libraries cannot have fallback functions. |  |  |
| typeError | "Fallback function must be payable or non-payable, but is \"" + stateMutabilityToString(fallback->stateMutability()) + "\"." |  |  |
| typeError | Fallback function cannot take parameters. |  |  |
| typeError | Fallback function cannot return values. |  |  |
| typeError | Fallback function must be defined as \"external\". |  |  |
| typeError | Function overload clash during conversion to external types for arguments. |  |  |
| typeError | Function signature hash collision for ") + it.second->externalSignature() |  |  |
| typeError | Library is not allowed to inherit. |  |  |
| typeError | Library cannot have non-constant state variables |  |  |
| typeError | Library is not allowed to inherit | [Line 476](https://github.com/ethereum/solidity/blob/efd8d8fe5eced023476af71491e9eae3dbde4d87/libsolidity/analysis/ContractLevelChecker.cpp#L476) | library BritishMuseum {  string constant name = "British Museum";  }  library MyLibrary is BritishMuseum {    } |
|  | Function has no declaration?! |  |  |
|  | Type only supported by the new experimental ABI encoder", currentLoc |  |  |
| fatalTypeError | "Contract \"") + \_contract.name() + "\" does not use the new experimental ABI encoder but wants to inherit from a contract " + "which uses types that require it. " + "Use \"pragma experimental ABIEncoderV2;\" for the inheriting contract as well to enable the feature." |  |  |

* **ContractFlowAnalyzer.cpp**

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Type** | **Message** | **Source File** |  |
| typeError | The variable was declared here.", variableOccurrence->declaration().location()  This variable is of storage pointer type and can be ") + (variableOccurrence->kind() == VariableOccurrence::Kind::Return ? "returned" : "accessed") + " without prior assignment." | [Lines 135 - 146](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ControlFlowAnalyzer.cpp#L135-L146)  **I really do not understand this error** | **function retrieve(address \_sender) internal view returns (mapping(address => uint) storage) {**    **}** |
| warning | Unreachable code. | [Line 179](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ControlFlowAnalyzer.cpp#L179) |  |

* **ContractFlowBuilder.cpp**
* *Nothing here, but might need to deep dive to understand what does this file do*
* **ContractFlowGraph.cpp**
* *Same*
* **DeclarationContainer.cpp**

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Type** | **Message** | **Source File** |  |
| solAssert | Tried to activate a non-inactive variable or multiple inactive variables with the same name. | [Line 85-88](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/DeclarationContainer.cpp#L85-L88) |  |
| solAssert | Attempt to update function definition. | [Line 113](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/DeclarationContainer.cpp#L113) |  |
| solAssert | Attempt to resolve empty name. | [Line 128](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/DeclarationContainer.cpp#L128) |  |

* **DocStringAnalyser.cpp**

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Type** | **Message** | **Source File** |  |
| DocstringParsingError | "Documented parameter \"" + i->second.paramName + "\" not found in the parameter list of the function." | [Line 87-92](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/DocStringAnalyser.cpp#L87-L92) |  |
| DocstringParsingError | Doc tag @ <tagName> not valid for " + \_nodeName + ". | [Line 134](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/DocStringAnalyser.cpp#L134) | /// @title sqrt()  /// @notice Math function to calculate square root  /// @dev Not working with decimal numbers  /// @param x The number to calculate the square root of  /// @return y The square root of x  function sqrt(uint x) internal pure returns (uint y) {  uint z = (x + 1) / 2;  y = x;  while (z < y) {  y = z;  z = (x / z + z) / 2;  }  } |
| DocstringParsingError | End of tag <tagName> not found | Can’t find it anywere |  |

* **GlobalContext.cpp**
* *Nothing*

**NameAndTypeResolver.cpp**

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Type** | **Message** | **Source File** |  |
| solAssert | Unable to register global declaration. | [Lines 50 - 53](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/NameAndTypeResolver.cpp#L50-L53) |  |
| declarationError | Import \"" + path + "\" (referenced as \"" + imp->path() + "\") not found. | [Lines 80 - 87](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/NameAndTypeResolver.cpp#L80-L88) |  |
| declarationError | "Declaration \"" + alias.first->name() + "\" not found in \"" + path + "\" (referenced as \"" + imp->path() + "\")." | [Lines 95 - 108](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/NameAndTypeResolver.cpp#L95-L108) |  |
| solAssert | Updated declaration outside global scope." | [Lines 143 - 147](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/NameAndTypeResolver.cpp#L143-L147) |  |
| solAssert | Found overloading involving something not a function, event or a (magic) variable. | [Lines 210 - 216](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/NameAndTypeResolver.cpp#L210-L216) |  |
| solAssert | Failed to determine the function type of the overloaded. | [Line 221](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/NameAndTypeResolver.cpp#L221) |  |
| fatalDeclarationError | Function type can not be used in this context. | [Lines 224 - 225](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/NameAndTypeResolver.cpp#L224-L225) |  |
| warning | Variable is shadowed in inline assembly by an instruction of the same name | [Lines 255 - 258](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/NameAndTypeResolver.cpp#L255-L258) |  |
| declarationError | "The previous declaration is here:", firstDeclarationLocation), “Identifier already declared.” | [Lines 365 - 369](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/NameAndTypeResolver.cpp#L365-L369) |  |
| fatalTypeError | Contract expected. | [Lines 382 - 383](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/NameAndTypeResolver.cpp#L382-L383) |  |
| fatalTypeError | Definition of base has to precede definition of derived contract | [Lines 388 - 389](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/NameAndTypeResolver.cpp#L388-L389) | pragma solidity ^0.5.0;  contract MyContract is MyLibrary {    function encodeString(string memory sentence) public pure returns (bytes memory) {  return abi.encode(sentence);  }    function decodeString(bytes memory data) public pure returns (string memory) {  return abi.decode(data, (string));  }    }  library MyLibrary {  // interface code here  } |
| fatalTypeError | Linearization of inheritance graph impossible | [Lines 394 - 395](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/NameAndTypeResolver.cpp#L394-L395) |  |
| solAssert | Scopes not correctly closed. | [Line 469](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/NameAndTypeResolver.cpp#L469) |  |
| declarationError | "The previous declaration is here:", firstDeclarationLocation), "Identifier already declared." | [Lines 515 - 519](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/NameAndTypeResolver.cpp#L515-L519) |  |
| warning | This declaration shadows a builtin symbol. | [Lines 524 - 528](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/NameAndTypeResolver.cpp#L524-L528) |  |
| warning | This declaration shadows an existing declaration. The shadowed declaration is here:", shadowedLocation | [Lines 529 - 537](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/NameAndTypeResolver.cpp#L529-L537) |  |
| solAssert | Variable declaration without function. | [Line 679](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/NameAndTypeResolver.cpp#L679) |  |
| solAssert | Unable to add new scope. | [Line 708](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/NameAndTypeResolver.cpp#L708) |  |
| solAssert | Closed non-existing scope. | [Line 714](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/NameAndTypeResolver.cpp#L714) |  |
| solAssert | No current scope. | [Line 720](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/NameAndTypeResolver.cpp#L720) |  |

**PostTypeChecker.cpp**

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Type** | **Message** | **Source File** |  |
| typeError | "The value of the constant " + declaration->name() +  " has a cyclic dependency via " + identifier->name() + "." | [Line 52 - 57](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/PostTypeChecker.cpp#L52-L57) |  |
| fatalDeclarationError | Variable definition exhausting cyclic dependency validator. | [Line 96 - 97](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/PostTypeChecker.cpp#L96-L97) |  |

**ReferencesResolver.cpp**

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Type** | **Message** | **Source File** |  |
| declarationError | Undeclared identifier. […]  - is not (or not yet) visible at this point.  - Did you mean … | [Line 99 - 111](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ReferencesResolver.cpp#L99-L111) | library MyLib {  function testCall() external {  test();  }    function test() public {  testCall();  }  } |
| typeError | Address types can only be payable or non-payable. | [Line 138 - 141](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ReferencesResolver.cpp#L138-L141) |  |
| fatalDeclarationError | Identifier not found or not unique. | [Line 176 - 180](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ReferencesResolver.cpp#L176-L180) |  |
| typeError | Name has to refer to a struct, enum or contract. | [Line 190 - 194](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ReferencesResolver.cpp#L190-L194) | address constant \_address1 = msg.sender;    function setTwoTx(\_address1, address payable \_address2) public {  \_address1.transfer(10);  \_address2.transfer(10);  }    function sendTx(address \_address) public {  setTwoTx(\_address);  } |
| fatalTypeError | Invalid visibility, can only be \"external\" or \"internal\". | [Line 199 - 207](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ReferencesResolver.cpp#L199-L207) |  |
| fatalTypeError | Only external function types can be payable. | [Line 209 - 213](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ReferencesResolver.cpp#L209-L213) |  |
| solAssert | Type not set for parameter | [Line 218](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ReferencesResolver.cpp#L218) |  |
| fatalTypeError | Internal type cannot be used for external function type. | [Line 219 - 223](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ReferencesResolver.cpp#L219-L223) |  |
| fatalTypeError | Illegal base type of storage size zero for array. | [Line 248 - 249](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ReferencesResolver.cpp#L248-L249) |  |
| fatalTypeError | Invalid array length, expected integer literal or constant expression. | [Line 256 - 257](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ReferencesResolver.cpp#L256-L257) |  |
| fatalTypeError | Array with zero length specified. | [Line 258 - 259](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ReferencesResolver.cpp#L258-L259) |  |
| fatalTypeError | Array with fractional length specified. | [Line 260 - 261](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ReferencesResolver.cpp#L260-L261) |  |
| fatalTypeError | Array with negative length specified. | [Line 262 - 263](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ReferencesResolver.cpp#L262-L263) |  |
| declarationError | In variable names \_slot and \_offset can only be used as a suffix. | [Lines 297 - 301](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ReferencesResolver.cpp#L297-L301) |  |
| declarationError | Multiple matching identifiers. Resolving overloaded identifiers is not supported. | [Lines 304 - 308](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ReferencesResolver.cpp#L304-L308) |  |
| declarationError | Cannot access local Solidity variables from inside an inline assembly function. | [Lines 312 - 316](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ReferencesResolver.cpp#L312-L316) |  |
| declarationError | The \"constant\" keyword can only be used for state variables. | [Lines 339 – 350](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ReferencesResolver.cpp#L349-L350) |  |
| typeError | Data location can only be specified for array, struct or mapping types | [Lines 381 - 382](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ReferencesResolver.cpp#L381-L382) |  |
| typeError | Data location must be /memory/storage/calldata/none for return / external for variable | [Lines 383 - 402](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ReferencesResolver.cpp#L383-L402) |  |
| solAssert | Data location not properly set. | [Lines 437 - 438](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ReferencesResolver.cpp#L437-L438) |  |

**ViewPureChecker.cpp**

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Type** | **Message** | **Source File** |  |
| warning | "Function state mutability can be restricted to " + stateMutabilityToString(m\_bestMutabilityAndLocation.mutability) | [Line 169 - 181](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ViewPureChecker.cpp#L169-L181) |  |
| typeError | "Function declared as pure, but this expression (potentially) reads from the environment or state and thus requires \"view\"." | [Line 252 - 265](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ViewPureChecker.cpp#L252-L265) |  |
| typeError | "Function declared as " + tateMutabilityToString(m\_currentFunction->stateMutability()) + ", but this expression (potentially) modifies the state and thus " "requires non-payable (the default) or payable." | [Line 266 - 276](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ViewPureChecker.cpp#L266-L276) |  |
| typeError | "\"msg.value\" or \"callvalue()\" appear here inside the modifier.", \*\_nestedLocation "This modifier uses \"msg.value\" or \"callvalue()\" and thus the function has to be payable or internal." | [Line 283 - 288](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ViewPureChecker.cpp#L283-L288) | modifier TxFee(uint \_fee) {  msg.value + \_fee;  \_;  }    function sendEther(address payable \_recipient, uint \_ether) public TxFee(10) {  \_recipient.transfer(\_ether);  } |
| typeError | "\"msg.value\" and \"callvalue()\" can only be used in payable public functions. Make the function " "\"payable\" or use an internal function to avoid this error." | [Line 289 - 294](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/ViewPureChecker.cpp#L289-L294) |  |

**Warnings**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Warning** | **Source code file + line** | **Other references** | **Code example** |
| warning | Unused function parameter. Remove or comment out the variable name to silence this warning. | StaticAnalyzer.cpp  [Line 124](https://github.com/ethereum/solidity/blob/efd8d8fe5eced023476af71491e9eae3dbde4d87/libsolidity/analysis/StaticAnalyzer.cpp#L124) |  | function sayHello(string memory \_name, uint8 \_age) public pure returns (string memory) {  string memory myName = \_name;  return myName;  } |
| warning | Unused local variable. | StaticAnalyzer.cpp  [Line 127](https://github.com/ethereum/solidity/blob/efd8d8fe5eced023476af71491e9eae3dbde4d87/libsolidity/analysis/StaticAnalyzer.cpp#L127) |  | function birthday(uint8 \_age) public pure returns (uint8) {  uint8 new\_age = \_age++;  return \_age++;  } |
| warning | Variable covers a large part of storage and thus makes collision likely. Either use mappings or dynamic arrays and allow their size to be increased only in small quantities per transaction | StaticAnalyzer.cpp  [Line 158 - 164](https://github.com/ethereum/solidity/blob/efd8d8fe5eced023476af71491e9eae3dbde4d87/libsolidity/analysis/StaticAnalyzer.cpp#L158-L164) | [*https://ethereum.stackexchange.com/questions/49278/initializing-full-array-warning*](https://ethereum.stackexchange.com/questions/49278/initializing-full-array-warning) |  |
| warning | Statement has no effect | StaticAnalyzer.cpp  [Line 185](https://github.com/ethereum/solidity/blob/1cc8475309dd1ae36436b0a5cb2285de0e679a35/libsolidity/analysis/StaticAnalyzer.cpp#L185) |  |  |
| warning | The constructor of the contract (or its base) uses inline assembly. Because of that, it might be that the deployed bytecode is different from type(...).runtimeCode. | StaticAnalyzer.cpp  [Line 213-214](https://github.com/ethereum/solidity/blob/1cc8475309dd1ae36436b0a5cb2285de0e679a35/libsolidity/analysis/StaticAnalyzer.cpp#L213-L214) |  |  |
| warning | \"this\" used in constructor. Note that external functions of a contract. cannot be called while it is being constructed." | StaticAnalyzer.cpp  [Line 234-240](https://github.com/ethereum/solidity/blob/1cc8475309dd1ae36436b0a5cb2285de0e679a35/libsolidity/analysis/StaticAnalyzer.cpp#L234-240) |  | constructor(uint \_ether) public {  this.sendEther(\_ether);  }    function sendEther(uint \_ether) external {  address(this).balance + \_ether;  msg.sender.balance - \_ether;  } |

*See here :*

**Errors**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Type** | **Message** | **Source code file + line** | **Other references** | **Code example** |
| typeError | \"msg.gas\" has been deprecated in favor of \"gasleft()\" | StaticAnalyzer.cpp  [line 195-199](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/StaticAnalyzer.cpp#L195-L199) |  |  |
| typeError | \"block.blockhash()\" has been deprecated in favor of \"blockhash()\" | StaticAnalyzer.cpp  [Lines 200-204](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/StaticAnalyzer.cpp#L200-L204) |  |  |
| typeError | \"callcode\" has been deprecated in favour of \"delegatecall\". | StaticAnalyzer.cpp  [Line 219 - 225](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/StaticAnalyzer.cpp#L219-L225) |  |  |
| **typeError** | Division by zero." : "Modulo zero. | StaticAnalyzer.cpp  [Line 284-288](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/StaticAnalyzer.cpp#L284-L288) |  | function divide(uint \_number) public pure {  uint result = \_number / 0;  } |
| **typeError** | Arithmetic modulo zero. | StaticAnalyzer.cpp  [Line 306-310](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/StaticAnalyzer.cpp#L306-L310) |  |  |
| **typeError** | The function declaration is here: <functionType->declaration().scope()->location()> Libraries cannot call their own functions externally. | StaticAnalyzer.cpp  [Line 312-324](https://github.com/ethereum/solidity/blob/f05805c955f73fd2ea1d14dc9edf14b472631b17/libsolidity/analysis/StaticAnalyzer.cpp#L312-L324) | Since Compiler version 0.5.9  <https://github.com/ethereum/solidity/issues/6451>  **https://github.com/ethereum/solidity/pull/6604/files** | pragma solidity 0.5.9;  library L {  using L for \*;    function f() public pure returns (uint r) {  return r.g();  }    function g(uint) public pure returns (uint) {  return 2;  }  } |

**SyntaxChecker.cpp**

The files related to the syntax checker are the SyntaxChecker.h, but most importantly /liblangutil/Token.h

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| SyntaxError | Source file does not specify required compiler version! Consider adding “pragma solidity” | [Line 57 - 72](https://github.com/ethereum/solidity/blob/2ee272acf32fbad4efd1da7919c59792597ce9e6/libsolidity/analysis/SyntaxChecker.cpp#L57-L72) | > error occurs here  contract NewHello{  // contract code here  } |
| SyntaxError | Invalid pragma | SyntaxChecker.cpp |  |
| SyntaxError | Experimental feature name is missing | [Line 86 - 90](https://github.com/ethereum/solidity/blob/1cc8475309dd1ae36436b0a5cb2285de0e679a35/libsolidity/analysis/SyntaxChecker.cpp#L86-L90) | pragma experimental;  contract MyContract {  // Write your code here  } |
| SyntaxError | Stray arguments | [Line 91 - 95](https://github.com/ethereum/solidity/blob/1cc8475309dd1ae36436b0a5cb2285de0e679a35/libsolidity/analysis/SyntaxChecker.cpp#L91-L95) | pragma experimental ABIEncoderV2 nextGen;  contract MyContract {  // Write your code here  } |
| SyntaxError | Empty experimental feature name is invalid. | SyntaxChecker.cpp |  |
| SyntaxError | Unsupported experimental feature name. | SyntaxChecker.cpp |  |
| SyntaxError | Duplicate experimental feature name. | [Line 103 - 104](https://github.com/ethereum/solidity/blob/1cc8475309dd1ae36436b0a5cb2285de0e679a35/libsolidity/analysis/SyntaxChecker.cpp#L103-L104) | pragma experimental SMTChecker;  pragma experimental SMTChecker;  contract MyContract {  // Write your code here  } |
| Warning | Experimental features are turned on. Do not use experimental features on live deployments. | SyntaxChecker.cpp | **See excel sheet** |
| SyntaxError | "Source file requires different compiler version (current compiler is " + string(VersionString) + " - note that nightly builds are considered to be strictly less than the released version" | SyntaxChecker.cpp |  |
| SyntaxError | Unknown pragma | SyntaxChecker.cpp | **pragma NewSolidity ^0.5.0;** |
| SyntaxError | Modifier body does not contain '\_' | [Line 143 - 144](https://github.com/ethereum/solidity/blob/1cc8475309dd1ae36436b0a5cb2285de0e679a35/libsolidity/analysis/SyntaxChecker.cpp#L143-L144) | pragma solidity ^0.5.9;  contract MyContract {    modifier Fee {  require (msg.value != 0);  }  } |
| SyntaxError | Variable declarations can only be used inside blocks. | [Line 151 - 152](https://github.com/ethereum/solidity/blob/1cc8475309dd1ae36436b0a5cb2285de0e679a35/libsolidity/analysis/SyntaxChecker.cpp#L151-L152) | pragma solidity ^0.5.9;  contract MyContract {    uint a = 3;  uint b = 2;    function test() public {    for (uint x = 1; x <= 10; x++) uint c = a + b + x;    }    } |
| SyntaxError | \"continue\" has to be in a \"for\" or \"while\" loop. | [Line 189 - 191](https://github.com/ethereum/solidity/blob/1cc8475309dd1ae36436b0a5cb2285de0e679a35/libsolidity/analysis/SyntaxChecker.cpp#L189-L191) | pragma solidity ^0.5.9;  contract MyContract {    uint a = 3;  uint b = 2;    function test(uint x) public {  if ( x < a && x > b) {  bool result = true;  continue;  }  }    } |
| SyntaxError | \"break\" has to be in a \"for\" or \"while\" loop. | Line 197 - 199 | pragma solidity ^0.5.9;  contract MyContract {    uint a = 3;  uint b = 2;    function test(uint x) public {  if ( x < a && x > b) {  bool result = true;  break;  }  }    } |
| SyntaxError | \"throw\" is deprecated in favour of \"revert()\", \"require()\" and \"assert()\" | SyntaxChecker.cpp |  |
| SyntaxError | Invalid use of underscores in number literal. No trailing underscores allowed. | SyntaxChecker.cpp | uint constant bitcoin\_supply = 21\_000\_000\_; |
| SyntaxError | Invalid use of underscores in number literal. Only one consecutive underscores between digits allowed. | SyntaxChecker.cpp | uint constant bitcoin\_supply = 21\_000\_\_000; |
| SyntaxError | Invalid use of underscores in number literal. No underscores in front of the fraction part allowed. | [Lines 236 - 237](https://github.com/ethereum/solidity/blob/1cc8475309dd1ae36436b0a5cb2285de0e679a35/libsolidity/analysis/SyntaxChecker.cpp#L236-L237) | fixed constant test = 10\_000\_.5; |
| SyntaxError | Invalid use of underscores in number literal. No underscores in front of the fraction part allowed. | [Lines 239 - 240](https://github.com/ethereum/solidity/blob/1cc8475309dd1ae36436b0a5cb2285de0e679a35/libsolidity/analysis/SyntaxChecker.cpp#L239-L240) | fixed constant test = 10\_000.\_5; |
| SyntaxError | Invalid use of underscores in number literal. No underscore at the end of the mantissa allowed. | SyntaxChecker.cpp |  |
| SyntaxError | Invalid use of underscores in number literal. No underscore in front of exponent allowed. | SyntaxChecker.cpp |  |
| SyntaxError | Use of unary + is disallowed. | SyntaxChecker.cpp |  |
| SyntaxError | The msize instruction cannot be used when the Yul optimizer is activated because it can change its semantics. Either disable the Yul optimizer or do not use the instruction. | SyntaxChecker.cpp |  |
| SyntaxError | Functions are not allowed to have the same name as the contract. If you intend this to be a constructor, use \"constructor(...) { ... }\" to define it. | SyntaxChecker.cpp | pragma solidity ^0.5.9;  contract MyContract {    function MyContract() public {  // do something  }    } |
| SyntaxError | No visibility specified. Did you intend to add \"" + suggestedVisibility + "\"? | SyntaxChecker.cpp | function test() {  // do something  } |
| SyntaxError | Functions without implementation cannot have modifiers. | SyntaxChecker.cpp | modifier TxFee(uint \_fee) {  msg.value + \_fee;  \_;  }    function sendEther() public TxFee; |
| Warning | Naming function type parameters is deprecated. | SyntaxChecker.cpp |  |
| SyntaxError | Return parameters in function types may not be named. | SyntaxChecker.cpp |  |
| SyntaxError | The use of the \"var\" keyword is disallowed. The declaration part of the statement can be removed, since it is empty | SyntaxChecker.cpp |  |
| SyntaxError | Defining empty structs is disallowed. | SyntaxChecker.cpp | struct Unknown {  // Struct definition here  } |

**TypeChecker.cpp (really long, 2500 lines of code)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Error Type** | **Message** | **Source File** |  |  |
|  | Type requested but not present. |  |  |  |
| Warning | "This assignment performs two copies to storage. Since storage copies do not first copy to a temporary location, one of them might be overwritten before the second is executed and thus may have unexpected effects. It is safer to perform the copies separately or assign to storage pointers first." |  |  |  |
| typeError | "This function takes two arguments, but " + toString(arguments.size()) + " were provided." |  |  |  |
| typeErrorConcatenateDescriptions | "Invalid type for argument in function call. Invalid implicit conversion from " + type(\*arguments.front())->toString() + " to bytes memory requested.", result.message() |  |  |  |
| typeError | "The second argument to \"abi.decode\" has to be a tuple of types." |  |  | function decodeNb(bytes memory data) public pure returns (uint8) {  uint8 nb;  nb = abi.decode(data, uint8);  return nb;  } |
| typeError | "Decoding type " + actualType->toString(false) + " not supported." |  |  |  |
| typeError | "Argument has to be a type name." |  |  |  |
| typeError | "This function takes one argument, but " + toString(arguments.size()) + " were provided." | [Line 209](https://github.com/ethereum/solidity/blob/efd8d8fe5eced023476af71491e9eae3dbde4d87/libsolidity/analysis/TypeChecker.cpp#L209) |  |  |
| typeError | "Invalid type for argument in function call. Contract type required, but " + type(\*arguments.front())->toString(true) + " provided." |  |  |  |
| solAssert | Base contract not available. |  |  |  |
| typeError | Interfaces cannot inherit. |  |  | pragma solidity ^0.5.0;  contract MyContract {    function encodeString(string memory sentence) public pure returns (bytes memory) {  return abi.encode(sentence);  }    function decodeString(bytes memory data) public pure returns (string memory) {  return abi.decode(data, (string));  }    }  interface MyInterface is MyContract {  // interface code here  } |
| typeError | Libraries cannot be inherited from. |  |  | pragma solidity ^0.5.0;  library MyLibrary {  // interface code here  }  contract MyContract is MyLibrary {    function encodeString(string memory sentence) public pure returns (bytes memory) {  return abi.encode(sentence);  }    function decodeString(bytes memory data) public pure returns (string memory) {  return abi.decode(data, (string));  }    } |
| typeError | "Wrong argument count for constructor call: " + toString(arguments->size()) + " arguments given but expected " + toString(parameterTypes.size()) + ". Remove parentheses if you do not want to provide arguments here." |  |  |  |
| typeErrorConcatenateDescriptions | "Invalid type for argument in constructor call. Invalid implicit conversion from " + type(\*(\*arguments)[i])->toString() + " to " + parameterTypes[i]->toString() + " requested.", |  |  |  |
| fatalTypeError | Library name expected. |  |  |  |
| solAssert | Type cannot be used in struct. |  |  |  |
| fatalDeclarationError | Struct definition exhausting cyclic dependency validator. |  |  |  |
| fatalTypeError | Recursive struct definition. |  |  | struct FootballPlayer {  string name;  uint8 age;  FootballPlayer mentor;  } |
| typeError | Library functions cannot be payable. |  |  |  |
| typeError | Internal functions cannot be payable. |  |  |  |
| typeError | Mapping types can only have a data location of \"storage\" and thus only be parameters or return variables for internal or library functions. |  |  |  |
| typeError | Mapping types can only have a data location of \"storage\". |  |  |  |
| TypeError | Mapping types for parameters or return variables can only be used in internal or library functions. |  |  | pragma solidity ^0.5.0;  contract myContract {    mapping(address => bool) votes;    function resetVotes(mapping(address => bool) memory \_voters) public pure {    }  } |
| typeError | Type is required to live outside storage. |  |  |  |
| solAssert / fatalTypeError | Expected detailed error message! |  |  |  |
| typeError | This type is only supported in the new experimental ABI encoder. Use \"pragma experimental ABIEncoderV2;\" to enable the feature. |  |  |  |
| declarationError | Base constructor already provided. |  |  |  |
| typeError | Functions in interfaces cannot have an implementation. |  |  |  |
| typeError | Functions in interfaces must be declared external. |  |  |  |
| typeError | Constructor cannot be defined in interfaces. |  |  |  |
| typeError | Constructor cannot be defined in libraries. |  |  |  |
| typeError | Constructor must be implemented if declared. |  |  |  |
| typeError | Internal library function must be implemented if declared. |  |  | **Wrong :**  library Messenger {    function testConnection() internal pure returns(bool);    }  **Right :**  library Messenger {    function testConnection() public pure returns(bool);    } |
| typeError | Variables cannot be declared in interfaces. |  |  | pragma solidity ^0.5.0;  interface LegalContract {    address party1;  address party2;    }  interface LegalContract2 {    struct Parties {  address party1;  address party2;  }  } |
| solAssert | Variable type not provided. |  |  |  |
| typeError | Constants of non-value type not yet implemented. |  |  |  |
| typeError | Uninitialized \"constant\" variable. |  |  | uint constant bitcoin\_supply; |
| typeError | Initial value for constant variable has to be compile-time constant. |  |  | pragma solidity ^0.5.0;  contract Example {    address constant \_address1 = msg.sender;    } |
| typeError | "Type " + varType->toString() + " is only valid in storage." |  |  |  |
| typeError | "The following types are only supported for getters in the new experimental ABI encoder: " + oinHumanReadable(unsupportedTypes) + ". Either remove \"public\" or use \"pragma experimental ABIEncoderV2;\" to enable the feature." |  |  |  |
| typeError | Internal or recursive type is not allowed for public state variables. |  |  |  |
| typeError | Array is too large to be encoded. |  | **If you go over 134\_217\_727, the error appears.**  **Apparently, this is the max limit of a working storage :**  <https://www-01.ibm.com/support/docview.wss?uid=swg21220835>  <https://www.ibm.com/support/knowledgecenter/en/SS6SG3_4.2.0/com.ibm.entcobol.doc_4.2/MG/igymapxg001.htm>  <http://ibmmainframes.com/about56564.html> | pragma solidity ^0.5.0;  contract MyContract {    uint[1000] \_large\_array;    function sayHello() internal pure returns (string memory) {  return "Hello";  }    function notPossibleToEncode() public pure returns (bytes memory) {  uint[134\_217\_728] memory large\_array;  return abi.encode(large\_array);  }  } |
| typeError | Referenced declaration is neither modifier nor base class. |  |  |  |
| typeError | "Wrong argument count for modifier invocation: " + toString(arguments.size()) + " arguments given but expected " + toString(parameters->size()) + "." |  |  |  |
| typeErrorConcatenateDescriptions | "Invalid type for argument in modifier invocation. " "Invalid implicit conversion from " + type(\*arguments[i])->toString() + " to " + type(\*(\*parameters)[i])->toString() + " requested." |  |  |  |
| typeError | Type is required to live outside storage. |  |  |  |
| typeError | Internal or recursive type is not allowed as event parameter type. |  |  |  |
| typeError | "This type is only supported in the new experimental ABI encoder. Use \"pragma experimental ABIEncoderV2;\" to enable the feature." |  |  |  |
| typeError | More than 4 indexed arguments for anonymous event. |  |  |  |
| typeError | More than 3 indexed arguments for event. |  |  |  |
| solAssert | External function type uses internal types. |  |  |  |
| solAssert | Expected variable type! |  |  |  |
| typeError | Constant variables not supported by inline assembly. |  |  |  |
| typeError | The suffixes \_offset and \_slot can only be used on storage variables. |  |  |  |
| typeError | Storage variables cannot be assigned to. |  |  |  |
| typeError | Only local variables are supported. To access storage variables, use the \_slot and \_offset suffixes. |  |  |  |
| typeError | You have to use the \_slot or \_offset suffix to access storage reference variables. |  |  |  |
| typeError | Call data elements cannot be accessed directly. Copy to a local variable first or use \"calldataload\" or \"calldatacopy\" with manually determined offsets and sizes. |  |  |  |
| typeError | Only types that use one stack slot are supported. |  |  |  |
| typeError | The suffixes \_offset and \_slot can only be used on storage variables. |  |  |  |
| typeError | Only local variables can be assigned to in inline assembly. | [**Line 676**](https://github.com/ethereum/solidity/blob/efd8d8fe5eced023476af71491e9eae3dbde4d87/libsolidity/analysis/TypeChecker.cpp#L676) |  |  |
| solAssert | Type of declaration required but not yet determined. |  |  |  |
| typeError | Expected a library. |  |  |  |
| typeError | Return arguments required |  |  |  |
| typeError | Return arguments not allowed. |  |  |  |
| typeError | Different number of arguments in return statement than in returns declaration. |  | **Here, the case is if we have multiple return value declared in the function head.**  Here we look at how many components are mentioned in the tuple in the final return statement.  We compare if this number **is different than the number of arguments mentioned in the return declaration** (function head) |  |
| typeErrorConcatenateDescriptions | "Return argument type " + type(\*\_return.expression())->toString() + " is not implicitly convertible to expected type " + TupleType(returnTypes).toString(false) + ".", result.message() |  |  |  |
| typeError | Different number of arguments in return statement than in returns declaration." |  | **Here, the case is if we have only one return value declared in the function head.**  Here we look at how many components are mentioned in the tuple in the final return statement. We check **if this number is not equal to one.** |  |
| typeErrorConcatenateDescriptions | "Return argument type " + type(\*\_return.expression())->toString() + " is not implicitly convertible to expected type (type of first return variable) " + expected->toString() + ".", result.message() |  |  |  |
| typeError | Expression has to be an event invocation. |  |  |  |
| fatalTypeError | Use of the \"var\" keyword is disallowed. | [Line 874](https://github.com/ethereum/solidity/blob/1cc8475309dd1ae36436b0a5cb2285de0e679a35/libsolidity/analysis/TypeChecker.cpp#L874)  [Line 879](https://github.com/ethereum/solidity/blob/1cc8475309dd1ae36436b0a5cb2285de0e679a35/libsolidity/analysis/TypeChecker.cpp#L879) |  |  |
| solAssert | Uninitialized storage pointer.  Expected a specified location at this point |  |  |  |
| typeError | Uninitialized mapping. Mappings cannot be created dynamically, you have to assign them from a state variable. |  |  |  |
| typeError | "Different number of components on the left hand side (" + toString(variables.size()) + ") than on the right hand side (" + toString(valueTypes.size()) + ")." |  |  |  |
| solAssert | Value has to be tied to statement |  |  |  |
| fatalTypeError | "Invalid rational " +valueComponentType->toString() +" (absolute value too large or division by zero)." |  |  |  |
| solAssert | Cannot declare variable with void (empty tuple) type. |  |  |  |
| solAssert | ", which can hold values between " + minValue + " and " + maxValue; | [**Line 974**](https://github.com/ethereum/solidity/blob/1cc8475309dd1ae36436b0a5cb2285de0e679a35/libsolidity/analysis/TypeChecker.cpp#L974) | **Not clear, need to deep dive and understand what this does** |  |
| solAssert | Unknown type. |  |  |  |
| typeError | "Type " + valueComponentType->toString() + " is not implicitly convertible to expected type " + var.annotation().type->toString(); + ", but it can be explicitly converted." |  |  |  |
| typeError | "Type " + valueComponentType->toString() + " is not implicitly convertible to expected type " + var.annotation().type->toString(); + ". Try converting to type " + valueComponentType->mobileType()->toString() + " or use an explicit conversion." |  |  |  |
| solAssert | Should have errors! |  |  |  |
| syntaxError | "Use of the \"var\" keyword is disallowed. Type cannot be expressed in syntax." |  |  |  |
| syntaxError | "Use of the \"var\" keyword is disallowed. Use explicit declaration `" + createTupleDecl(variables) + " = ...´ instead." |  |  |  |
| typeError | Invalid rational number. |  |  |  |
| warning | Return value of low-level calls not used. |  |  |  |
| warning | Failure condition of 'send' ignored. Consider using 'transfer' instead. |  |  |  |
| typeError | Invalid mobile type in true expression. |  |  |  |
| typeError | Invalid mobile type in false expression. |  |  |  |
| typeError | "True expression's type " + trueType->toString() + " doesn't match false expression's type " + falseType->toString() + "." |  |  |  |
| typeError | Conditional expression as left value is not supported yet. |  | **Related to LValue** |  |
| solAssert | Array sizes don't match or no errors generated. |  |  |  |
| typeError | Mappings cannot be assigned to. |  |  |  |
| typeError | Compound assignment is not allowed for tuple types. |  |  |  |
| typeError | "Operator " + string(TokenTraits::toString(\_assignment.assignmentOperator())) + " not compatible with types " + t->toString() + " and " + type(\_assignment.rightHandSide())->toString() |  |  |  |
| fatalTypeError | Inline array type cannot be declared as LValue. |  |  |  |
| fatalTypeError | Tuple component cannot be empty. |  |  |  |
| fatalTypeError | Array component cannot be empty. |  |  |  |
| typeError | Tuple component cannot be empty. |  |  |  |
| fatalTypeError | Invalid rational number. |  |  |  |
| solAssert | Inline array cannot have empty components |  |  |  |
| fatalTypeError | Invalid mobile type. |  |  |  |
| fatalTypeError | Unable to deduce common type for array elements. |  |  |  |
| fatalTypeError | "Type " + inlineArrayType->toString() + " is only valid in storage." |  |  |  |
| typeError | "Unary operator " + string(TokenTraits::toString(op)) + " cannot be applied to type " + subExprType->toString() |  |  |  |
| typeError | "Operator " + string(TokenTraits::toString(\_operation.getOperator())) + " not compatible with types " + leftType->toString() + " and " + rightType->toString() + (!result.message().empty() ? ". " + result.message() : "") |  |  |  |
| warning | "Result of exponentiation / shift has type " + commonType->toString() + " and thus might overflow. Silence this warning by converting the literal to the expected type." |  |  |  |
| typeError | Exactly one argument expected for explicit type conversion. |  |  |  |
| typeError | Type conversion cannot allow named arguments. |  |  |  |
| solAssert | Invalid explicit conversion to storage type. |  |  |  |
| ??? | Did you mean to declare this variable as "address payable"? |  |  |  |
| typeError | "Explicit type conversion not allowed from non-payable \"address\" to \"" + resultType->toString() + "\", which has a payable fallback function." |  |  |  |
| typeError | "Explicit type conversion not allowed from \"" + argType->toString() + "\" to \"" + resultType->toString() + "\"." |  |  |  |
| typeError | "staticcall" is not supported by the VM version. |  |  |  |
| typeError | Event invocations have to be prefixed by \"emit\". |  |  |  |
| solAssert | ABI function has unexpected FunctionType::Kind. |  | From here, ABI related |  |
| solAssert | ABI functions should be variadic. |  |  |  |
| solAssert | ABI function with unexpected padding |  |  |  |
| typeError | Named arguments cannot be used for functions that take arbitrary parameters. |  |  |  |
| typeError | Fractional numbers cannot yet be encoded. |  |  |  |
| typeError | Invalid rational number (too large or division by zero). |  |  |  |
| typeError | Cannot perform packed encoding for a literal. Please convert it to an explicit type first. |  |  |  |
| typeError | Type not supported in packed mode. |  |  |  |
| typeError | This type cannot be encoded. |  |  |  |
| solAssert | Struct constructor calls cannot be variadic. |  | From here, general checks |  |
| typeError | "Need at least " + toString(parameterTypes.size()) + " arguments for " + string(isStructConstructorCall ? "struct constructor" : "function call") + ", but provided only " + toString(arguments.size()) + "." |  |  |  |
| typeError | "Wrong argument count for " + string(isStructConstructorCall ? "struct constructor" : "function call") + ": " + toString(arguments.size()) + " arguments given but " + string(isVariadic ? "need at least " : "expected ") + toString(parameterTypes.size()) + "."; |  |  | function setTwoTx(address payable \_address1, address payable \_address2) public {  \_address1.transfer(10);  \_address2.transfer(10);  }    function sendTx(address \_address) public {  setTwoTx(\_address);  } |
| typeError | Members that have to be skipped in memory: <member> |  |  |  |
| typeError | " This function requires a single bytes argument. Use \"\" as argument to provide empty calldata." |  |  |  |
| typeError | " This function requires a single bytes argument. If all your arguments are value types, you can use abi.encode(...) to properly generate it." |  |  |  |
| typeError | " This function requires a single bytes argument. Use abi.encodePacked(...) to obtain the pre-0.5.0 behaviour or abi.encode(...) to use ABI encoding." |  |  |  |
| solAssert | Unexpected parameter length mismatch! |  |  |  |
| typeError | "Duplicate named argument \"" + \*argumentNames[i] + "\"." |  |  |  |
| typeError | "Named argument \*argumentNames[i] does not match function declaration." |  |  |  |
| solAssert | unmapped parameter |  |  |  |
| typeError | "Invalid type for argument in function call. Invalid implicit conversion from <type(\*paramArgMap[i])->toString()> to <parameterTypes[i]->toString()> requested." |  |  | pragma solidity ^0.5.0;  contract Example {    string \_age\_in\_string = "25";    function HappyBirthday(uint8 \_age) public pure returns (uint8) {  return \_age + 1;  }    function NotWorking() public pure {  HappyBirthday(\_age\_in\_string);  }  } |
| typeError | " This function requires a single bytes argument. If all your arguments are value types, you can use abi.encode(...) to properly generate it." |  |  |  |
| typeError | " This function requires a single bytes argument. Use abi.encodePacked(...) to obtain the pre-0.5.0 behaviour or abi.encode(...) to use ABI encoding." |  |  |  |
| typeError | Type is not callable |  | **?????** |  |
| solAssert | Type name not resolved. |  |  |  |
| fatalTypeError | Identifier is not a contract. |  |  |  |
| fatalTypeError | Cannot instantiate an interface. |  |  |  |
| typeError | Missing implementation: function->location(). Trying to create an instance of an abstract contract. |  |  |  |
| typeError | Contract with internal constructor cannot be created directly. |  |  |  |
| solAssert | Linearized base contracts not yet available. |  |  |  |
| typeError | Circular reference for contract creation (cannot create instance of derived or same contract). |  |  |  |
| fatalTypeError | Type cannot live outside storage. |  |  |  |
| typeError | Length has to be placed in parentheses after the array type for new expression. |  |  |  |
| fatalTypeError | Contract or array type expected. |  |  |  |
| fatalTypeError | Member <memberName> is not available in <exprType->toString()> outside of storage. |  |  |  |
| ??? | Member <memberName> not found or not visible after argument-dependent lookup in <exprType->toString()>. Did you intend to call the function? |  |  |  |
|  | Constructor for " + t.front()->toString() + " must be payable for member \"value\" to be available. |  |  |  |
|  | "Member \"value\" is not allowed in delegated calls due to \"msg.value\" persisting." |  |  |  |
|  | "Member \"value\" is only available for payable functions." |  |  |  |
|  | Member <memberName> not found or not visible after argument-dependent lookup in <exprType->toString()>. Use \"address(" + varName + ")." + memberName + "\" to access this address member. |  |  |  |
| solAssert | Expected address not-payable as members were not found |  |  |  |
|  | "send" and "transfer" are only available for objects of type "address payable", not <exprType->toString()>". |  |  |  |
| fatalTypeError | Member <memberName> not unique after argument-dependent lookup in <exprType->toString()> - did you forget the "payable" modifier? |  | [**Line 2116-2118**](https://github.com/ethereum/solidity/blob/1cc8475309dd1ae36436b0a5cb2285de0e679a35/libsolidity/analysis/TypeChecker.cpp#L2116) |  |
| fatalTypeError | Member <memberName> not unique after argument-dependent lookup in <exprType->toString |  | [**Line 2116-2118**](https://github.com/ethereum/solidity/blob/1cc8475309dd1ae36436b0a5cb2285de0e679a35/libsolidity/analysis/TypeChecker.cpp#L2116) |  |
| solAssert | Function <memberName> cannot be called on an object of type <exprType->toString()> (expected <funType->selfType()->toString()>). |  |  |  |
| typeError | Circular reference for contract code access. |  |  |  |
| typeError | Index expression cannot be omitted. |  |  |  |
| typeError | Index access for string is not possible. |  |  |  |
| typeError | Out of bounds array access. |  |  |  |
| typeError | Index expression cannot be omitted. |  |  |  |
| typeError | Index access for contracts or libraries is not possible. |  |  |  |
| fatalTypeError | Integer constant expected. |  |  |  |
| solAssert | Expected errors as expectType returned false |  |  |  |
| typeError | Index expression cannot be omitted. |  |  |  |
| fatalTypeError | Index expression cannot be represented as an unsigned integer. |  |  |  |
| typeError | Out of bounds array access. |  |  |  |
| fatalTypeError | Indexed expression has to be a type, mapping or array (is <baseType->toString()>) |  |  |  |
| fatalTypeError | No matching declaration found after variable lookup. |  |  |  |
| fatalTypeError | No unique declaration found after variable lookup. |  |  |  |
| fatalTypeError | No candidates for overload resolution found. |  |  |  |
| solAssert | Requested type not present. |  |  |  |
| fatalTypeError | No matching declaration found after argument-dependent lookup. |  |  |  |
| fatalTypeError | No unique declaration found after argument-dependent lookup. |  |  |  |
| solAssert | Referenced declaration is null after overload resolution. |  |  |  |
| fatalTypeError | Declaration referenced before type could be determined. |  |  |  |
| typeError | "sha3" has been deprecated in favour of "keccak256" |  |  |  |
| typeError | "suicide" has been deprecated in favour of "selfdestruct" |  |  |  |
| syntaxError | "This looks like an address but is not exactly 40 hex digits. It is <to\_string(\_literal.valueWithoutUnderscores().length() - 2)> hex digits." |  |  |  |
| syntaxError | This looks like an address but has an invalid checksum. Correct checksummed address: <\_literal.getChecksummedAddress>. |  |  |  |
| syntaxError | If this is not used as an address, please prepend '00'  For more information please see https://solidity.readthedocs.io/en/develop/types.html#address-literals | [**Line 2400**](https://github.com/ethereum/solidity/blob/1cc8475309dd1ae36436b0a5cb2285de0e679a35/libsolidity/analysis/TypeChecker.cpp#L2400) |  |  |
| fatalTypeError | Hexadecimal numbers cannot be used with unit denominations. You can use an expression of the form "0x1234 \* 1 day" instead. |  |  |  |
| typeError | Using \"years\" as a unit denomination is deprecated. |  |  |  |
| fatalTypeError | Invalid literal value. |  |  |  |
| solAssert | Declaration not stored. |  |  |  |
| solAssert | Declaration not stored. |  |  |  |
| typeError | Type <type(\_expression)->toString()> is not implicitly convertible to expected type <\_expectedType.toString()>, but it can be explicitly converted  . |  |  |  |
| typeError | Type <type(\_expression)->toString()> is not implicitly convertible to expected type <\_expectedType.toString()>. Try converting to type <type(\_expression)->mobileType()->toString()  > or use an explicit conversion. |  |  |  |
| typeError | Cannot assign to a constant variable. |  |  |  |
| typeError | Expression has to be an lvalue. |  |  |  |

**There are also a lot of files to analyse in the path** */solidity/libsolidity/codegen*

* *ABIFunctions.cpp*
* *ArrayUtils.cpp*
* *Compiler.cpp (Nothing)*
* *CompilerContext.cpp*
* *CompilerUtils.cpp*
* *ContractCompiler.cpp*
* *ExpressionCompiler.cpp*
* *LValue.cpp*

## ***MultiUseYulFunctionCollector.cpp***

## ***YulUtilFunctions.cpp***

***LValue.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Message** | **Description** | **Source** | **Code snippet example** |
| *Errinfo\_comment* | *Stack too deep, try removing local variables* | *Line 48* |  |
| *solAssert* | *Size and stack pos mismatch* | *Line 50* |  |
| *Errinfo\_comment* | *Stack too deep, try removing local variables* | *Line 62* |  |
| *solAssert* | *Invalid non-padded type* | *Line 110* |  |
| *solUnimplementedAssert* | Conversion not implemented for assignment to memory. | *Line 129* |  |
| *solAssert* | Invalid storage size | *Line 158* |  |
| *solAssert* | Invalid storage ref size | *Line 167* |  |
| *solUnimplemented* | Not yet implemented - FixedPointType | *Line 186* |  |
| *solAssert* | Invalid storage bytes size | *Line 224-225* |  |
| *solAssert* | Invalid stack size | *Line 228* |  |
| *solAssert* | Function item stored but target is not equal to source | *Line 258* |  |
| *solAssert* | Source not fixed bytes | *Line 269* |  |
|  | Invalid stack size for opaque type | *Line 274* |  |
| *solAssert* | Wrong type conversation for assignment | *Line 289* |  |
| *solAssert* | Struct assignment with conversion | *Line 308* |  |
| *solAssert* | Struct in calldata not supported | *Line 311* |  |
| *solAssert* | Unexpected Source size | *Line 347* |  |
| *Errinfo\_comment / internal compilerError* | Invalid non-value type for assignment. | *Line 357* |  |
| *solAssert* | "Clearing of unsupported type requested: " + m\_dataType->toString() | *Line 392* |  |

***CompilerContext.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Message** | **Description** | **Source** | **Code snippet example** |
| *solAssert* | "Invalid stack height in low-level function " + name + "." | *Line 122* |  |
| *solAssert* | Compiled contract not found. | *Line 176, line 183* |  |
| *solAssert* | No inheritance hierarchy set. | *Line 210,*  *line 217,*  *line 245*  *Line 486* |  |
| *solAssert* | Function modifier " + \_modifier.name() + " not found in inheritance hierarchy. | *Line 250* |  |
| *solAssert* | Variable not found on stack. | *Line 256* |  |
| *solAssert* | Variable not found in storage. | *Line 274* |  |
|  |  | *Lines 306 – 318 are a bit harder to understand (assembly)* |  |
| *CompilerError* | Stack too deep (" + to\_string(stackDiff) + "), try removing local variables. | *Line 373* |  |
| *solAssert* | Error parsing/analyzing inline assembly block:\n  ------------------ Input: -----------------  ------------------ Errors: -----------------  ------------------------------------------- | *Lines 395 - 403* |  |
| *reportError* | Invalid assembly generated by code generator. | *Line 419* |  |
| *reportError* | Failed to analyze inline assembly block. | *Line 447,*  *Line 449* |  |
| *solAssert* | Super function " + name + " not found. | *Line 480* |  |
| *solAssert* | Base not found in inheritance hierarchy. | *Line 488* |  |

***ArrayUtils.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Message** | **Description** | **Source** | **Code snippet example** |
| *solAssert* | "Byte offset for array as base type. | *Line 186* |  |
| *solAssert* | Byte offset for direct copy. | *Line 197* |  |
| *solUnimplemented* | Copying of type " + \_sourceType.toString(false) + " to storage not yet supported. | *Line 220* |  |
| *solAssert* | Stack too deep, try removing local variables. | *Line 224* |  |
| solUnimplementedAssert | Nested dynamic arrays not implemented here. | *Line 294* |  |
| *solAssert* | Invalid storage size for non-value type. | *Line 546*  *Line 650*  *Line 792*  *Line 841* |  |
| *solAssert* | Invalid storage size for type. | *Line 547* |  |
| *solAssert* | Invalid size for value type. | *Line 550* |  |
| *solAssert* | Invalid storage size. | *Line 570* |  |
| *solAssert* | Invalid statically sized non-value base type on array access. | *Line 1173* |  |

***ContractCompiler.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Message** | **Description** | **Source** | **Code snippet example** |
| *solAssert* | I sense a disturbance in the stack: ") + to\_string(m\_context.stackHeight()) + " vs " + to\_string(stackHeight) | *Line 61* |  |
| *solAssert* | Tried to initialize library. | *Line 130* |  |
| *solAssert* | Tried to use contract creator or library. | *Line 152* |  |
| *solAssert* | Runtime sub not registered | *Line 168*  *Line 185* |  |
| *solAssert* | Tried to deploy contract as library. | *Line 181* |  |
| *solAssert* | CALL / DELEGATECALL flag expected. | *Line 357* |  |
| *solAssert* | Libraries can't have fallback functions | *Line 361* |  |
| *solAssert* | Tried to initialize state variables of library. | *Line 479* |  |
| *solAssert* | Compiler visit to non-state variable declaration. | *Line 487* |  |
| *CompilerError* | Stack too deep, try removing local variables. | *Line 557*  *Line 711* |  |
| *solAssert* | Invalid stack layout on cleanup. | *Line 572* |  |
| *solAssert* | Type of declaration required but not yet determined. | *Line 610* |  |
| *solAssert* | Invalid constant in inline assembly. | *Line 668* |  |
| *solAssert* | Invalid declaration type. | *Line 726* |  |
| *solAssert* | Can only assign to stack variables in inline assembly. | *Line 736* |  |
| *compilerError* | Stack too deep(" + to\_string(stackDiff) + "), try removing local variables. | *Line 744* |  |
| *solAssert* | Invalid return parameters pointer. | *Line 897* |  |
| *solAssert* | Throw statement is disallowed. | *Line 921* |  |
| *solAssert* | Compiled the wrong function? | *Line 1011* |  |
|  |  |  |  |

***ABIFunctions.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Type** | **Message** | **Source** | **Code snippet example** |
| *solUnimplementedAssert* | Encoding type \"" + \_to.toString() + "\" not yet implemented. | *Line 266* |  |
| *solAssert* | Non-padded / inplace encoding for library call requested. | *Line 339* |  |
| *solAssert* | Invalid encoded size. | *Line 386* |  |
| *solAssert* | Unknown dynamic type. | *Line 408 - 409* |  |
| *solUnimplementedAssert* | Encoding type \"" + member.type->toString() + "\" not yet implemented. | *Line 840* |  |

**CompilerUtils.cpp**

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Type** | **Message** | **Source** | **Code snippet example** |
| *staticAssert* | Free memory pointer must not overlap with scratch area. | Line 44 |  |
| *staticAssert* | Zero pointer must not overlap with free memory pointer. | Line 45 |  |
| *staticAssert* | General purpose memory must not overlap with zero area. | Line 46 |  |
| *solAssert* | Invalid dynamically encoded base type on tail access. | Line 120 |  |
| *solAssert* | Invalid dynamically sized type. | Line 126 |  |
| *solAssert* | Unable to statically load dynamic type. | Line 152 |  |
| *SolUnimplementedAssert* | Only in-memory reference type can be stored. | Line 201 |  |
| *solAssert* | Memory store of type " + \_type.toString(true) + " not allowed. | Line 234 |  |
| *SolUnimplementedAssert* | No decoding type found. | Line 267 |  |
| *SolUnimplementedAssert* | Nested arrays not yet implemented. | Line 271 |  |
| *SolUnimplementedAssert* | Nested memory arrays not yet implemented here. | Line 276 |  |
| *solAssert* | Unknown dynamically sized type: " + type->toString() | Line 370 |  |
| *solUnimplementedAssert* | Encoding type \"" + t->toString() + "\" not yet implemented. | Line 396 |  |
| *solAssert* | Non-padded and in-place encoding can only be combined. | Line 407 |  |
| *solAssert* | Externalable type expected. | Line 429  Line 440  Line 473 |  |
| *solAssert* | Stack too deep, try using fewer variables. | Line 435 |  |
| *solAssert* | Unknown dynamic type. | Line 492 |  |
| *solAssert* | Library calls cannot be packed. | Line 529 |  |
| *solUnimplemented*  *solUnimplementedAssert* | Not yet implemented - FixedPointType. | Line 712  Line 765  Line 815  Line 833 |  |
| *solAssert* | Invalid type conversion requested. | Line 736  Line 1128 |  |
| *solAssert* | empty enum should have caused a parser error. | Line 754  Line 797 |  |
| *solAssert* | Invalid conversion to FixedBytesType requested. | Line 776 |  |
| *solAssert* | Invalid conversion to EnumType requested. | Line 792 |  |
| *solAssert* | Invalid conversion to FixedMxNType requested. | Line 808 |  |
| *solAssert* | Invalid conversion from string literal to " + \_targetType.toString(false) + " requested. | Line 884 |  |
| *solAssert* | Invalid conversion to storage type. | Line 900 |  |
| *solAssert* | Invalid conversion to calldata type. | Line 967 |  |
| *solAssert* | Invalid conversion to storage type. | Line 988 |  |
| *solAssert* | Member not found in target type. | Line 1014 |  |
| *solAssert* | Invalid type conversion target location CallData. | Line 1046 |  |
| *solAssert* | Invalid conversion for bool. | Line 1099 |  |
| *solAssert* | Only external function type can be converted. | Line 1107 |  |
| *solAssert* | Invalid function type conversion requested. | Line 1123 |  |
| *solAssert* | enum overflow checking missing. | Line 1138 |  |
| *solAssert* | forgot to chop the sign bits. | Line 1139 |  |
| *solAssert* | Requested initialisation for unknown type: " + type->toString() | Line 1206 |  |
| *solAssert* | Variable size and position mismatch. | Line 1223 |  |
| *CompilerError* | Stack too deep, try removing local variables. | Line 1227  Line 1237 |  |
| *solAssert* | Static memory load of more than 32 bytes requested. | Line 1362 |  |
| *solAssert* | Memory store of types with stack size != 1 not allowed (Type: " + \_type.toString(true) + ") | Line 1421 |  |
| *solAssert* | Memory store of 0 bytes requested (Type: " + \_type.toString(true) + ") | Line 1428 |  |
| *solAssert* | Memory store of more than 32 bytes requested (Type: " + \_type.toString(true) + ") | Line 1433 |  |

***ExpressionCompiler.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Type** | **Message** | **Source** | **Code snippet example** |
| *solAssert* | **Type information not available.** | **Line 58** |  |
| *solAssert* | **Expected string or byte array for mapping key type** | **Line 116** |  |
| *solAssert* | **Expected value type for mapping key** | **Line 140** |  |
| *solAssert* | **Index access is allowed only for \"mapping\" and \"array\" types.** | **Line 165** |  |
| *CompilerError* | **Stack too deep.** | **Line 217** |  |
| *solAssert* | LValue not retrieved. | **Line 270**  **Line 378**  **Line 384** |  |
| *solAssert* | Compound operators only available for value types. | **Line 276** |  |
| *CompilerError* | **Stack too deep, try removing local variables.** | **Line 301** |  |
| *solAssert* | **Cannot create dynamically sized inline array.** | **Line 319** |  |
| *solUnimplementedAssert* | **Not yet implemented - FixedPointType.** | **Line 387**  **Line 1767** |  |
| *solUnimplementedAssert* | **Stack size != 1 not implemented.** | **Line 393** |  |
| *solAssert* | **Invalid unary operator: " + string(TokenTraits::toString(\_unaryOperation.getOperator()** | **Line 420** |  |
| *solAssert* | Callcode has been removed. | **Line 612** |  |
| *solAssert* | Gas limit set for contract creation. | **Line 616** |  |
| *solAssert* | **Too many indexed arguments.** | **Line** |  |
| *solAssert* | **\_memberAccess has no type** | **Line 1182** |  |
| *solAssert* | Function not found in member access | **Line 1196** |  |
| *solAssert* | event not found | **Line 1200** |  |
| *solAssert* | **unsupported member function** | **Line 1224** |  |
| *solAssert* | **Contract member is neither variable nor function.** | **Line 1261**  **Line 1291** |  |
| *solAssert* | **Referenced declaration not resolved.** | **Line 1276** |  |
| *solAssert* | Invalid member access in contract | **Line 1296** |  |
| *solAssert* | Invalid member access to integer | **Line 1301** |  |
| *solAssert* | **Invalid member access to address** | **Line 1331** |  |
| *solAssert* | **Invalid member access to function.** | **Line 1344** |  |
| *solAssert* | Gas has been removed. | **Line 1373** |  |
| *solAssert* | Blockhash has been removed. | **Line 1376** | No that can’t be true ! that’s such a great technology ! ☹ |
| *solAssert* | **Unknown magic member.** | **Line 1409** |  |
| *solAssert* | **Illegal data location for struct.** | **Line 1464** |  |
| *solAssert* | Tried to use ." + member + "() on a non-dynamically sized array | **Line 1504** |  |
| *solAssert* | Illegal array member. | **Line 1508** |  |
| *solAssert* | Illegal fixed bytes member. | **Line 1518** |  |
| *solAssert* | **Member access to unknown type.** | **Line 1522** |  |
| *solAssert* | **Index expression expected.** | **Line 1538**  **Line 1569**  **Line 1598** |  |
| *solAssert* | **Index access to string not allowed** | **Line 1580** |  |
| *solAssert* | **Index access only allowed for mappings or arrays.** | **Line 1619** |  |
| *soLAssert* | **Identifier type not expected in expression context.** | **Line 1672** |  |
| *solUnimplemented* | Only integer, boolean and string literals implemented for now. | **Line 1691** |  |
| *solAssert* | Comparison of multi-slot types. | **Line 1712** |  |
| *solAssert* | Unknown comparison operator. | **Line 1755** |  |
| *solAssert* | Unknown binary operator. | **Line 1767** |  |
| *solAssert* | **Unknown arithmetic operator.** | **Line 1806** |  |
| *solAssert* | Unknown bit operator. | **Line 1824** |  |
| *solAssert* | Only integer and fixed bytes type supported for shifts. | **Line 1836** |  |
| *solAssert* | **Invalid shift amount type.** | **Line 1849** |  |
| *solAssert* | **Unknown shift operator.** | **Line 1903** |  |
| *solAssert* | **Callcode has been removed.** | **Line 1939** |  |
| *solAssert* | Value set for delegatecall | **Line 2067** |  |
| *solAssert* | Value set for staticcall | **Line 2069** |  |
| *solUnimplementedAssert* | **Not implemented for non-value types.** | **Line 2209** |  |
| *InternalCompilerError* | **Identifier type not supported or identifier not found.** | **Line 2235** |  |

## ***MultiUseYulFunctionCollector.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Type** | **Message** | **Source** | **Code snippet example** |
| *solAssert* | **Function not properly named** | **Line 48** |  |

## ***MultiUseYulFunctionCollector.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Type** | **Message** | **Source** | **Code snippet example** |
| *solAssert* | **Asserts can't have messages!** | **Line 114** |  |
| *solAssert* | Left align requested for rational number. | **Line 188** |  |
| *solUnimplemented* | Fixed point types not implemented. | **Line 193** |  |
| *solAssert* | Left align requested for non-value type. | **Line 198** |  |
| *solAssert* | **Left align requested for inaccessible dynamic type.** | **Line 213** |  |
| *solAssert* | **Left align of type " + \_type.identifier() + " requested.** | **Line 216** |  |
| *solAssert* | **called regular array length function on calldata array** | **Line 549** |  |
| *solUnimplementedAssert* | **Byte Arrays not yet implemented!** |  |  |
| *solUnimplemented* | **Calldata arrays not yet implemented!** | **Line 785** |  |
| *solAssert* | Invalid storage bytes size. | **Line 935** |  |
| *solAssert* | Invalid storage bytes size. | **Line 936** |  |
| *solAssert* | Invalid non-value type for assignment. | **Line 961** |  |
| *solAssert* | Can only update types with location memory. | **Line 978** |  |
| *solAssert* | Memory store of type " + \_type.toString(true) + " not allowed. | **Line 1018** |  |
| *solUnimplementedAssert* | Not yet implemented - FixedPointType. | **Line 1195**  **Line 1220** |  |
| *solAssert* | Invalid conversion to FixedBytesType requested. | **Line 1200** |  |
| *solAssert* | Invalid conversion for bool. | **Line 1259** |  |
| *solUnimplemented* | Fixed point types not implemented. | **Line 1267**  **Line 1380** |  |
| *solUnimplementedAssert* | Array conversion not implemented. | **Line 1285** |  |
| *solUnimplementedAssert* | Struct conversion not implemented. | **Line 1290** |  |
| *solAssert* | Conversion should not be called for function types. | **Line 1319** |  |
| *solUnimplementedAssert* | Tuple conversion not implemented. | **Line 1334** |  |
| *solAssert* | Cleanup requested for non-storage reference type. | **Line 1389** |  |
| *solAssert* | empty enum should have caused a parser error. | **Line 1469** |  |
| *solAssert* | Expected signed type! | **Line 1599** |  |
| *solUnimplementedAssert* | Stacksize not yet implemented! | **Line 1620** |  |
| *solUnimplementedAssert* | Zero value for non-value types not yet implemented" | **Line 1621** |  |
| *solUnimplementedAssert* | Type conversion " + \_from.toString() + " -> " + \_to.toString() + " not yet implemented. | **Line 1647** |  |
| *solAssert* | Invalid conversion from string literal to " + \_to.toString() + " requested. | **Line 1698** |  |

**Next :** */solidity/libsolidity/ast*

* *AST.cpp*
* *ASTAnnotations.cpp (Nothing)*
* *ASTJsonConverter.pp*
* *ASTPrinter.cpp (Nothing)*
* *TypeProvider.cpp*
* *Types.cpp*

***AST.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Type** | **Message** | **Source** | **Code snippet example** |
| ***solAssert*** | **Hash collision at Function Definition Hash calculation** | **Line 122** |  |
| *solAssert* | **addInheritableMember got a nullpointer.** | **Line 221** |  |
| *solAssert* | Enclosing Scope of EnumValue was not set | **Line 278** |  |
| *solAssert* | Enclosing Scope of FunctionDefinition was not set. | **Line 297** |  |
| *solAssert* | **visibility() should not return Default** | **Line 308**  **Line 576** |  |
| *solAssert* | **Can only be called after reference resolution** | **Line 522**  **Line 552** |  |
| *solAssert* | **Expected hex number** | **Line 677**  **Line 683** |  |

***ASTJsonConverter.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Type** | **Message** | **Source** | **Code snippet example** |
| **solAssert** | Unknown kind of function call. | **Line 789** |  |
| **solAssert** | Unknown kind of literal token. | **Line 805** |  |

***TypeProvider.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Type** | **Message** | **Source** | **Code snippet example** |
| **solAssert** | Expected an elementary type name but got " + \_type.toString() | **Line 207** |  |
| **solAssert** | Unable to convert elementary typename " + \_type.toString() + " to type. | **Line 246** |  |
| **solAssert** | Cannot parse elementary type: " + \_name | **Line 255** |  |
| **solAssert** | Unknown data location: " + nameParts[1] | **Line 274** |  |
| **solAssert** | Invalid state mutability for address type: " + nameParts[1] | **Line 285** |  |
| **solAssert** | Storage location suffix only allowed for reference types | **Line 291** |  |
| **solAssert** | MetaType is handled separately | **Line 528** |  |
| **solAssert** | Only contracts supported for now. | **Line 534** |  |

***Types.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Type** | **Message** | **Source** | **Code snippet example** |
| **Err** | Cannot implicitly convert signed literal to unsigned type. |  |  |
| **err** | Literal is too large to fit in " + \_type.toString(false) + ". |  |  |
| **err** | **Parameter should have external type.** |  |  |
| **Errinfo\_comment** | **Object too large for storage.** | **Line 190**  **Line 204** |  |
| **solAssert** | Identifier cannot start with a number. | **Line 305** |  |
| **solAssert** | Identifier contains invalid characters. | **Line 308** |  |
| **err** | **Arithmetic operations on addresses are not supported. Convert to integer first before using them.** | **Line 451** |  |
| **solAssert** | **Invalid bit number for integer type: " + dev::toString(m\_bits)** | **Line 504** |  |
| **Err** | Exponentiation is not allowed for signed integer types. | **Line 616** |  |
| **solAssert** | Invalid bit number(s) for fixed type: " + | **Line 629** |  |
| **err** | **Too many fractional digits.** | **Line 645** |  |
| **err** | **Precision of rational constants is limited to 4096 bits.** | **Line 1042**  **Line 1123** |  |
| **???** | **"...(" + to\_string(omitted) + " digits omitted)..." + str.substr(str.size() - 4, 4** | **Line 1155** |  |
| **solAssert** | Rational number cannot be represented as fixed point type. | **Line 1183** |  |
| **solAssert** | Number constant too large. | **Line 1189** |  |
| **solAssert** | Number constant too small. | **Line 1190** |  |
| **solAssert** | **integerType() called for fractional number.** | **Line 1209** |  |
| **???** | **literal\_string (contains invalid UTF-8 sequence at position " + dev::toString(invalidSequence) + ")** | **Line 1303** |  |
| **solAssert** | **Invalid byte number for fixed bytes type: " + dev::toString(m\_bytes)** | **Line 1322** |  |
| **solAssert** | **Bool type constructed from non-boolean literal.** | **Line 1400** |  |
| **solAssert** | **Array size does not fit unsigned.** | **Line 1667** |  |
| **Errinfo\_comment** | **Array too large for storage.** | **Line 1693** |  |
| **solAssert** | **Expected detailed error message!** | **Line 1815**  **Line 2139** |  |
| **solAssert** | **Array size does not fit u256.** | **Line 1840** |  |
| **solAssert** | **linearizedBaseContracts should at least contain the most derived contract.** | **Line 1889** |  |
| **solAssert** | **Override changes type.** | **Line 1904** |  |
| **solAssert** | Struct member not found. | **Line 2027** |  |
| **solAssert** | Parameter should have external type. | **Line 2035** |  |
| **err** | **Invalid type!** | **Line 2111** |  |
| **err** | **Recursive structs can only be passed as storage pointers to libraries, not as memory objects to contract functions.** | **Line 2131** |  |
| **???** | **Recursive type not allowed for public or external contract functions.** | **Line 2148** |  |
| **solAssert** | **Parameter should have external type.** | **Line 2191** |  |
| **solAssert** | **Storage offset of non-existing member requested.** | **Line 2228** |  |
| **solAssert** | **Member not found in struct.** | **Line 2240** |  |
| **solAssert** | **Requested unknown enum value " + \_member** | **Line 2323** |  |
| **solAssert** | **Storage size of non-storable tuple type requested.** | **Line 2372** |  |
| **solAssert** | Parameter names list must match parameter types list! | **Line 2439**  **Line 2505**  **Line 2526**  **Line 2565** |  |
| **solAssert** | Return parameter names list must match return parameter types list! | **Line 2444**  **Line 2509**  **Line 2530**  **Line 2569** |  |
| **solAssert** | Internal payable function type used. | **Line 2541** |  |
| **solAssert** | Type not set for parameter. | **Line 2544** |  |
| **solAssert** | Internal payable function type used. | **Line 2558** |  |
| **solAssert** | Storage size of non-storable function type requested. | **Line 2794** |  |
| **solAssert** | Alignment property of non-exportable function type requested. | **Line 2802**  **Line 2812** |  |
| **solAssert** | **Declaration needed to determine interface function type.** | **Line 2857** |  |
| **err** | **Internal type is not allowed for public or external functions.** | **Line 2959** |  |
| **solAssert** | **Expected equal sized type & name vectors** | **Line 2991** |  |
| **solAssert** | External signature of function needs declaration | **Line 3074** |  |
| **solAssert** | Fallback function has no signature. | **Line 3075** |  |
| **solAssert** | Invalid function type for requesting external signature. | **Line 3084** |  |
| **solAssert** | **Declaration has to be available.** | **Line 3174** |  |
| **solAssert** | Function is not bound. | **Line 3198** |  |
| **solAssert** | Function has no self type. | **Line 3199** |  |
| **solAssert** | Must be an elementary type! | **Line 3262** |  |
| **solAssert** | Expected detailed error message! | **Line 3270** |  |
| **err** | Only libraries are allowed to use the mapping type in public or external functions. | **Line 3275** |  |
| **solAssert** | Storage size of non-storable type type requested. | **Line 3295**  **Line 3362** |  |
| **solAssert** | Only contracts supported for now | **Line 3529** |  |
| **solAssert** | Unknown kind of magic | **Line 3542**  **Line 3562** |  |

**Next :** */solidity/libsolidity/formal*

* *BMC.cpp*
* *CHC.cpp*
* *CVC4Interface.cpp*
* *EncodingContext.cpp*
* *ModelChecker.cpp*
* *SMTEncoder.cpp*
* *SMTLib2Interface.cpp*
* *SMTPortfolio.cpp*
* *SSAVariable.cpp*
* *SymbolicTypes.cpp*
* *SymbolicVariables.cpp*
* *VariableUsage.cpp*
* *Z3CHCInterface.cpp*
* *Z3Interface.cpp*

**Next :** */solidity/libsolidity/interface*

* *ABI.cpp*
* *CompilerStack.cpp*
* *GasEstimator.cpp (Nothing)*
* *Natspec.cpp*
* *StandardCompiler.cpp*
* *Version.cpp (Nothing)*

***ABI.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Type** | **Message** | **Source** | **Code snippet example** |
| **solAssert** | **Names and types vector size does not match** | **Line 133** |  |
| **solAssert** | **Invalid type.** | **Line 197** |  |

***CompilerStack.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| ***Error Type*** | ***Message*** | ***Source*** | ***Code Snippet example*** |
| *solAssert* | You shall not have another CompilerStack aside me. | Line 88 |  |
| *Errinfo\_comment* | Must set remappings before parsing. | Line 121 |  |
| *Errinfo\_comment* | Must set EVM version before parsing. | Line 129 |  |
| *Errinfo\_comment* | Must set libraries before parsing. | Line 136 |  |
| *Errinfo\_comment* | Must set optimiser settings before parsing. | Line 151 |  |
| *Errinfo\_comment* | Must set use literal sources before parsing. | Line 158 |  |
| *Errinfo\_comment* | Must add SMTLib2 responses before parsing. | Line 165 |  |
| *Errinfo\_comment* | Cannot change sources once set. | Line 196 |  |
| *Errinfo\_comment* | Must set sources before parsing. | Line 198 |  |
| *Errinfo\_comment* | Must call parse only after the SourcesSet state. | Line 207 |  |
| *Warning* | This is a pre-release compiler version, please do not use it in production. | Line 212 |  |
| *warning* | "The Yul optimiser is still experimental. "  "Do not use it in production unless correctness of generated code is verified with extensive tests." | Line 216-217 |  |
| *solAssert* | Parser returned null but did not report error. | Line 230 |  |
| *Errinfo\_comment* | Must call analyze only after parsing was successful. |  |  |
| *Errinfo\_comment* | Parsing was not successful. | Line 463  Line 473 |  |
| *Errinfo\_comment* | Compilation was not successful. | Line 486  Line 495  Line 504  Line 518  Line 556 -> 610  Line 1354 |  |
| *Errinfo\_comment* | No compiled contracts found. | Line 532 |  |
| *Errinfo\_comment* | Analysis was not successful. | Line 639 -> 724 |  |
| *Errinfo\_comment* | No sources set. | Line 738 |  |
| *solAssert* | Import path cannot be empty. | Line 822 |  |
| *???* | File not supplied initially. | Line 833 |  |
| *parserError* | Source \"" + importPath + "\" not found: " + result.responseOrErrorMessage | Line 841 – 843 |  |
| *solAssert* | Optimizer exception during compilation | Line 965 |  |
| *solAssert* | Assembly exception for bytecode | Line 975 |  |
| *solAssert* | Assembly exception for deployed bytecode | Line 985 |  |
| *Errinfo\_comment* | Contract \"" + \_contractName + "\" not found. | Line 1067 |  |
| *Errinfo\_comment* | Given source file not found. | Line 1074 |  |
| *solAssert* | Scanner not available | Line 1098 |  |
| *Static\_assert* | Invalid word size. | Line 1110 |  |
| *solAssert* | Metadata too large. | Line 1193 |  |
| *solAssert* | Too many map entries. | Line 1194 |  |
| *solAssert* | Text string too large. | Line 1220 |  |
| *solAssert* | Byte string too large. | Line 1236 |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

***StandardCompiler.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| ***Error Type*** | ***Message*** | ***Source*** | ***Code Snippet example*** |
| *formatFatalError* | "JSONError", "\"" + \_name + "\" must be an object" | ***Line 323*** |  |
| *formatFatalError* | "JSONError", "Unknown key \"" + member + "\"" | ***Line 327*** |  |
| *formatFatalError* | "JSONError", "\"settings.optimizer.details." + \_name + "\" must be Boolean" | ***Line 373*** |  |
| *formatFatalError* | "JSONError", "\"settings.metadata.useLiteralContent\" must be Boolean" | ***Line 382*** |  |
| *formatFatalError* | ***"JSONError", "\"settings.outputSelection\" must be an object"*** | ***Line 390*** |  |
| *formatError* | ***"JSONError", "\"settings.outputSelection." + sourceName + "\" must be an object"*** | ***Line 398 – 399*** |  |
| *formatError* | ***"JSONError",***  ***"\"settings.outputSelection." +***  ***sourceName +***  ***"." +***  ***contractName +***  ***"\" must be a string array"*** | ***Line 408 - 413*** |  |
| *formatFatalError* | JSONError", "The \"enabled\" setting must be a Boolean. | ***Line 443*** |  |
| *formatFatalError* | JSONError", "The \"runs\" setting must be an unsigned number. | ***Line 451*** |  |
| *formatFatalError* | ***JSONError", "\"Providing yulDetails requires Yul optimizer to be enabled.*** | ***Line 480*** |  |
| *formatFatalError* | JSONError", "Input is not a JSON object. | ***Line 498*** |  |
| *formatFatalError* | JSONError", "\"sources\" is not a JSON object. | ***Line 508*** |  |
| *formatFatalError* | JSONError", "No input sources specified. | ***Line 511*** |  |
| *formatErroor* | ***"IOError", "general", "Mismatch between content and supplied hash for \"" + sourceName + "\""*** | ***Line 531 – 533***  ***Line 556 - 558*** |  |
| *formatFatalError* | ***JSONError", "No import callback supplied, but URL is requested.*** | ***Line 541*** |  |
| *formatFatalError* | ***JSONError", "URL must be a string.*** | ***Line 549*** |  |
| *???* | ***Cannot import url (\"" + url.asString() + "\"): " + result.responseOrErrorMessage*** | ***Line 568*** |  |
| *formatFatalError* | ***JSONError", "Invalid input source specified.*** | ***Line 583*** |  |
| *formatFatalError* | JSONError", "\"auxiliaryInput.smtlib2responses\" must be an object. | ***Line 597*** |  |
| *formatFatalError* | JSONError", "Invalid hex encoding of SMTLib2 auxiliary input | ***Line 608*** |  |
| *formatFatalError* | ***"JSONError",***  ***"\"smtlib2Responses." + hashString + "\" must be a string."*** | ***Line 613 - 614*** |  |
| *formatFatalError* | JSONError", "\"settings.parserErrorRecovery\" must be a Boolean. | ***Line 630*** |  |
| *formatFatalError* | JSONError", "evmVersion must be a string. | ***Line 637*** |  |
| *formatFatalError* | JSONError", "Invalid EVM version requested. | ***Line 640*** |  |
| *formatFatalError* | JSONError", "\"settings.remappings\" must be an array of strings. | ***Line 645*** |  |
| *formatFatalError* | JSONError", "\"settings.remappings\" must be an array of strings | ***Line 650*** |  |
| *formatFatalError* | JSONError", "Invalid remapping: \"" + remapping.asString() + "\" | ***Line 654*** |  |
| *formatFatalError* | JSONError", "\"libraries\" is not a JSON object. | ***Line 668*** |  |
| *formatFatalError* | JSONError", "Library entry is not a JSON object. | ***Line 673*** |  |
| *formatFatalError* | JSONError", "Library address must be a string. | ***Line 677*** |  |
| *formatFatalError* | ***"JSONError",***  ***"Library address is not prefixed with \"0x\"."*** | ***Line 682 – 683*** |  |
| *formatFatalError* | ***"JSONError",***  ***"Library address is of invalid length."*** | ***Line 688 - 689*** |  |
| *formatFatalError* | ***"JSONError",***  ***"Invalid library address (\"" + address + "\") supplied."*** | ***Line 700 - 701*** |  |
| *?????* | ***General*** | ***Line 763*** |  |
|  | ***General***  ***Uncaught error:*** | ***Line 775 – 776*** |  |
|  | ***"FatalError",***  ***"general",***  ***"Uncaught fatal error: " + boost::diagnostic\_information(\_exception)*** | ***Line 784 - 786*** |  |
|  | ***"CompilerError",***  ***"general",***  ***"Compiler error (" + \_exception.lineInfo() + ")"*** | ***Line 794 – 796*** |  |
|  | ***"InternalCompilerError",***  ***"general",***  ***"Internal compiler error (" + \_exception.lineInfo() + ")"*** | ***Line 804 - 806*** |  |
|  | ***"UnimplementedFeatureError",***  ***"general",***  ***"Unimplemented feature (" + \_exception.lineInfo() + ")"*** | ***Line 814 - 816*** |  |
|  | ***"Exception",***  ***"general",***  ***"Exception during compilation: " + boost::diagnostic\_information(\_exception)*** | ***Line 832 - 834*** |  |
|  | ***"Exception",***  ***"general",***  ***"Unknown exception during compilation."*** | ***Line 841 - 843*** |  |
|  | ***"InternalCompilerError", "No error reported, but compilation failed."*** | ***Line 852***  ***Line 982*** |  |
| *formatFatalError* | JSONError", "Yul mode only supports exactly one input file. | ***Line 962*** |  |
| *formatFatalError* | JSONError", "Yul mode does not support smtlib2responses. | ***Line 964*** |  |
| *formatFatalError* | JSONError", "Field \"settings.remappings\" cannot be used for Yul. | ***Line 966*** |  |
| *formatFatalError* | JSONError", "Field \"settings.libraries\" cannot be used for Yul. | ***Line 968*** |  |
| *formatError* | ***Warning", "general", "Yul is still experimental. Please use the output with care.*** | ***Line 1005*** |  |
| *formatFatalError* | JSONError", "Only \"Solidity\" or \"Yul\" is supported as a language. | ***Line 1050*** |  |
| *formatFatalError* | InternalCompilerError", string("JSON logic exception: ") + \_exception.what() | ***Line 1054*** |  |
| *formatFatalError* | InternalCompilerError", string("JSON runtime exception: ") + \_exception.what() | ***Line 1058*** |  |
| *formatFatalError* | InternalCompilerError", "Internal exception in StandardCompiler::compile: " + boost::diagnostic\_information(\_exception) | ***Line 1062*** |  |
| *formatFatalError* | InternalCompilerError", "Internal exception in StandardCompiler::compile | ***Line 1066*** |  |
| *?????* | ***JSONError*** ***{\"errors\":[{\"type\":\"JSONError\",\"component\":\"general\",\"severity\":\"error\",\"message\":\"Error parsing input JSON.\"}]}*** | ***Line 1081*** |  |
|  | ***JSONError*** ***{\"errors\":[{\"type\":\"JSONError\",\"component\":\"general\",\"severity\":\"error\",\"message\":\"Error writing output JSON.\"}]}*** | ***Line 1094*** |  |
| *?????* |  |  |  |

**Next :** */solidity/libsolidity/parsing*

* *DocStringParser.cpp*
* *Parser.cpp*

***DocStringParser.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| ***Error Type*** | ***Message*** | ***Source*** | ***Code Snippet example*** |
|  | ***End of tag " + string(tagPos, tagNameEndPos) + " not found*** | ***Line 100*** |  |
|  | ***No param name given*** | ***Line 143*** |  |
|  | ***No description given for param " + paramName*** | ***Line 154*** |  |

***Parser.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| ***Error Type*** | ***Message*** | ***Source*** | ***Code Snippet example*** |
| *fatalParserError* | ***Expected pragma, import directive or contract/interface/library definition.*** | ***Line 100*** |  |
| *fatalParserError* | ***"Source file requires different compiler version (current compiler is " +***  ***string(VersionString) + " - note that nightly builds are considered to be "***  ***"strictly less than the released version"*** | ***Line 123 - 125*** |  |
| *parserError* | ***Token incompatible with Solidity parser as part of pragma directive.*** | ***Line 143*** |  |
| *fatalParserError* | Expected string literal (path), \"\*\" or alias list. | ***Line 219*** |  |
| *fatalParserError* | Expected \"from\". | ***Line 223*** |  |
| *fatalParserError* | Expected import path. | ***Line 226*** |  |
| *fatalParserError* | Import path cannot be empty. | ***Line 230*** |  |
| *solAssert* | ***Invalid contract kind.*** | ***Line 251*** |  |
| *fatalParserError* | ***Function, variable, struct or modifier declaration expected.*** | ***Line 312*** |  |
| *solAssert* | ***Invalid visibility specifier.*** | ***Line 376*** |  |
| *parserError* | ***"The state mutability modifier \"constant\" was removed in version 0.5.0. "***  ***"Use \"view\" or \"pure\" instead."*** | ***Line 400 - 401*** |  |
| *solAssert* | ***Invalid state mutability specifier.*** | ***Line 405*** |  |
| *solAssert* | ***Function or constructor expected.*** | ***Line 421*** |  |
| *fatalParserError* | ***"This function is named \"constructor\" but is not the constructor of the contract. "***  ***"If you intend this to be a constructor, use \"constructor(...) { ... }\" without the \"function\" keyword to define it."*** | ***Line 430 - 431*** |  |
| *parserError* | ***"Visibility already specified as \"" +***  ***Declaration::visibilityToString(result.visibility) +***  ***"\"."*** | ***Line 469 – 473***  ***Line 650 - 652*** |  |
| *parserError* | ***"State mutability already specified as \"" +***  ***stateMutabilityToString(result.stateMutability) +***  ***"\"."*** | ***Line 484 - 486*** |  |
| *fatalParserError* | ***Expected identifier after ','*** | ***Line 608*** |  |
| *parserError* | ***enum with no members is not allowed.*** | ***Line 611*** |  |
| *parserError* | Location already specified. | ***Line 668*** |  |
| *parserError* | Location specifier needs explicit type name. | ***Line 670*** |  |
| *solAssert* | ***Unknown data location.*** | ***Line 685*** |  |
| *parserError* | State mutability can only be specified for address types. | ***Line 882*** |  |
| *parserError* | Expected explicit type name. | ***Line 891*** |  |
| *fatalParserError* | Expected type name | ***Line 901*** |  |
| *solAssert* | ***Tried to parse type as constructor.*** | ***Line 914*** |  |
| *fatalParserError* | ***Expected elementary type name for mapping key type*** | ***Line 932*** |  |
| *fatalParserError* | ***Unexpected trailing comma in parameter list.*** | ***Line 964*** |  |
| *fatalParserError* | ***Only \"evmasm\" supported.*** | ***Line 1097*** |  |
| *fatalParserError* | ***Expected event name or path.*** | ***Line 1204*** |  |
| *parserError* | ***Expected expression (inline array elements cannot be omitted).*** | ***Line 1640*** |  |
| *fatalParserError* | ***Expected primary expression.*** | ***Line 1669*** |  |
| *parserError* | ***Unexpected trailing comma.*** | ***Line 1716*** |  |

**Files from** */libdevcore*

* CommonData.cpp (Nothing)
* CommonIO.cpp (Nothing)
* Exceptions.cpp (Nothing)
* *IndentedWritter.cpp*
* IPFSHash.cpp
* JSON.cpp
* Keccak256.cpp (nothing)
* StringUtils.cpp (nothing)
* SwarmHash.cpp (nothing)
* UTF8.cpp (nothing)
* Whiskers.cpp

***IndentedWritter.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| **ErrorType** | **Message** | **Source** | **Code Snippet Example** |
| ***assertThrow*** | Negative indentation. | ***Line 51*** |  |

***IpfsHash.cpp***

*Errinfo\_comment :* Ipfs hash for large (chunked) files not yet implemented. (Line 61)

***JSON.cpp***

Static\_assert : "Unexpected jsoncpp version: " JSONCPP\_VERSION\_STRING ". Expecting 1.8.4." **(line 36)**

***Whiskers.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| **ErrorType** | **Message** | **Source** | **Code Snippet Example** |
| *assertThrow* | **Parameter" + \_parameter + " contains invalid characters.** | **Line 79** |  |
| *assertThrow* | already set as value parameter. | **Line 88** |  |
| *assertThrow* | already set as condition parameter. | **Line 93** |  |
| *assertThrow* | already set as list parameter. | **Line 98** |  |
| *assertThrow* | **\_parameters.count(tagName),**  **WhiskersError,**  **"Value for tag " + tagName + " not provided.\n" +**  **"Template:\n" +**  **\_template** | **Line 119 - 123** |  |
| *assertThrow* | List parameter " + listName + " not set. | **Line 130** |  |
| *assertThrow* | Condition parameter " + conditionName + " not set. | **Line 142** |  |
| *assertThrow* | **Parameter collision** | **Line 166** |  |

***Files from /liblangutil***

List of files :

|  |  |  |  |
| --- | --- | --- | --- |
| **Status** | **Header files** | **Status** | Standard files |
| ***Listed*** | *CharStream.cpp* | **Nothing** | CharStream.h |
|  |  | **nothing** | Common.h |
| **nothing** | *EVMVersion.cpp (nothing)* | **nothing** | EVMVersion.h |
| **Listed** | *ErrorReporter.cpp* | **Listed** | *ErrorReporter.h* |
| **Listed** | *Exceptions.cpp* | **Listed** | *Exceptions.h* |
| **Listed** | *ParserBase.cpp* | **Nothing** | *ParserBase.h* |
| **Listed** | *Scanner.cpp* | **Listed** | *Scanner.h* |
| **Listed** | *SemVerHandler.cpp* | **Nothing** | *SemverHandler.h* |
|  |  |  | *SourceLocation.h* |
| **Nothing** | [SourceReferenceExtractor.cpp](https://github.com/ethereum/solidity/blob/develop/liblangutil/SourceReferenceExtractor.cpp) | **Nothing** | [SourceReferenceExtractor.h](https://github.com/ethereum/solidity/blob/develop/liblangutil/SourceReferenceExtractor.h) |
| **Listed** | [SourceReferenceFormatter.cpp](https://github.com/ethereum/solidity/blob/develop/liblangutil/SourceReferenceFormatter.cpp) | **Nothing** | [SourceReferenceFormatter.h](https://github.com/ethereum/solidity/blob/develop/liblangutil/SourceReferenceFormatter.h) |
| **Listed** | [SourceReferenceFormatterHuman.cpp](https://github.com/ethereum/solidity/blob/develop/liblangutil/SourceReferenceFormatterHuman.cpp) | **Nothing** | [SourceReferenceFormatterHuman.h](https://github.com/ethereum/solidity/blob/develop/liblangutil/SourceReferenceFormatterHuman.h) |
| **Listed** | [Token.cpp](https://github.com/ethereum/solidity/blob/develop/liblangutil/Token.cpp) | **Nothing, but most interesting file ☺** | [Token.h](https://github.com/ethereum/solidity/blob/develop/liblangutil/Token.h) |
| **Listed** | [UndefMacros.h](https://github.com/ethereum/solidity/blob/develop/liblangutil/UndefMacros.h) |  |  |

***CharStreamcpp***

|  |  |  |  |
| --- | --- | --- | --- |
| **ErrorType** | **Message** | **Source** | **Code Snippet Example** |
| ***solAssert*** | Attempting to set position past end of source. | ***Line 78*** |  |

***ErrorReporter.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| **ErrorType** | **Message** | **Source** | **Code Snippet Example** |
| ***Errinfo\_comment*** | There are more than 256 warnings. Ignoring the rest. | ***Line 103*** |  |
| ***Errinfo\_comment*** | There are more than 256 errors. Aborting. | ***Line 117*** |  |

***ErrorReporter.h***

|  |  |  |  |
| --- | --- | --- | --- |
| **ErrorType** | **Message** | **Source** | **Code Snippet Example** |
| ***solAssert*** | Need error descriptions ! | ***Line 97*** |  |

***Exceptions.cpp***

This file basically list the type of errors that the Solidity compiler will throw after compilation of the contract code.

***Exceptions.h***

" Truncated from " + std::to\_string(occurrences) + " to the first 32 occurrences."

Line 126

***ParserBase.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| **ErrorType** | **Message** | **Source** | **Code Snippet Example** |
|  | identifier | ***Line 63*** |  |
|  | End of source | ***Line 65*** |  |
|  | "reserved keyword '" + TokenTraits::friendlyName(\_token) + "'" |  |  |
| ***parserError / fatalParserError*** | "Expected " + expectedToken + " but got " + tokenName(tok) | ***Line 84***  ***Line 86***  ***(not sure if it’s correct or not)*** | ***pragma solidity ^0.4.0;***  ***contract Test {***    ***enum NUMBERS { ZERO, ONE, TWO }***    ***uint nb1;***  ***string nb2;***    ***function sendMoney(address payable recipient, uint \_value) external payable {***  ***if (5 > \_value) {***  ***recipient.transfer(\_value);***  ***}***  ***}***  ***}*** |
| ***fatalParserError*** | "In " + \_currentNodeName + ", " + expectedToken + "is expected; got " + ParserBase::tokenName(tok) + " instead." | ***Line 109*** |  |
| ***parserWarning*** | "Recovered in " + \_currentNodeName + " at " + expectedToken + "." | ***Line 120***  ***Line 129*** |  |
| ***parserError*** | "Recovered at next " + expectedToken | ***Line 122*** |  |
| ***fatalParserError*** | Maximum recursion depth reached during parsing | ***Line 141*** |  |

***Scanner.cpp***

*See Excel sheet file*

***Scanner.h***

CharStream memory locations must match. (Line 169)

***SemVerHandler.cpp***

*Invalid SemVer expression (solAssert, line 124)*

***SourceReferenceFormatter.cpp* (nb: the same is mentioned in SOurceReferenceFormatterHuman.cpp)**

|  |  |  |  |
| --- | --- | --- | --- |
| **ErrorType** | **Message** | **Source** | **Code Snippet Example** |
|  | ^ (Relevant source part starts here and spans across multiple lines). | **Line 63** |  |

***SourceReferenceFormatterHuman.cpp***

"^ (Relevant source part starts here and spans across multiple lines).\n" (line 116)

"Note" (line 130)

***Token.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| **ErrorType** | **Message** | **Source** | **Code Snippet Example** |
| **solAssert** | **"Expected elementary type name: " + string(TokenTraits::toString(\_baseType))** | **Line 54** |  |
| **solAssert** | **"There should not be a second size argument to type bytesM."** | **Line 57** |  |
| **solAssert** | "No elementary type bytes" + to\_string(\_first) + "." | **Line 58** |  |
| **solAssert** | "There should not be a second size argument to type " + string(TokenTraits::toString(\_baseType)) + "." | **Line 62** |  |
| **solAssert** | "No elementary type " + string(TokenTraits::toString(\_baseType)) + to\_string(\_first) + "." | **Line 65** |  |
| **solAssert** | "No elementary type " + string(TokenTraits::toString(\_baseType)) + to\_string(\_first) + "x" + to\_string(\_second) + "." | **Line 72** |  |

***Token.h***

***This is the most interesting file of the entire compiler. It list all the keywords and identifiers available in Solidity.***

"Token name size should be greater than 3. Should not reach here." (line 363)

***UndefMacros.h***

|  |
| --- |
| "The preceding macros in this header file are reserved for V8's "\ |
|  | "TOKEN\_LIST. Please add a platform specific define above to undefine "\ |
|  | "overlapping macros." |

***Lines 42 - 44***

***Files from /liblll***

|  |  |  |  |
| --- | --- | --- | --- |
| **Status** | **Header files** | **Status** | Standard files |
| **Not listed** | [CodeFragment.cpp](https://github.com/ethereum/solidity/blob/develop/liblll/CodeFragment.cpp) | **Not listed** | [CodeFragment.h](https://github.com/ethereum/solidity/blob/develop/liblll/CodeFragment.h) |
| **Not listed** | [Compiler.cpp](https://github.com/ethereum/solidity/blob/develop/liblll/Compiler.cpp) | **Not listed** | [Compiler.h](https://github.com/ethereum/solidity/blob/develop/liblll/Compiler.h) |
| **Not listed** | [CompilerState.cpp](https://github.com/ethereum/solidity/blob/develop/liblll/CompilerState.cpp) | **Not listed** | [CompilerState.h](https://github.com/ethereum/solidity/blob/develop/liblll/CompilerState.h) |
|  |  | **Not listed** | [Exceptions.h](https://github.com/ethereum/solidity/blob/develop/liblll/Exceptions.h) |
| **Not listed** | [Parser.cpp](https://github.com/ethereum/solidity/blob/develop/liblll/Parser.cpp) | **Not listed** | [Parser.h](https://github.com/ethereum/solidity/blob/develop/liblll/Parser.h) |

**Files from** */libevmasm*

* *Assembly.cpp*
* *AssemblyItem.cpp*
* *BlockDeduplicator.cpp (Nothing)*
* [CommonSubexpressionEliminator.cpp](https://github.com/ethereum/solidity/blob/develop/libevmasm/CommonSubexpressionEliminator.cpp)
* *ConstantOptimiser.cpp*
* *ControlFlowGraph.cpp*
* *ExpressionClasses.cpp*
* *GasMeter.cpp*
* *Instruction.cpp*
* *JumpDestRemover.cpp*
* *KnownState.cpp*
* *LinkerObject.cpp (Nothing)*
* *PathGasMeter.cpp (Nothing)*
* *PeepholeOptimiser.cpp*
* *SemanticInformation.cpp (Nothing)*
* *SimplificationRules.cpp*

***Assembly.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| **ErrorType** | **Message** | **Source** | **Code Snippet Example** |
| *assertThrow* | **Stack underflow.** | **Line 84** |  |
| *assertThrow* | **Empty named tag.** | **Line 353** |  |
| *assertThrow* | **Peephole optimizer seems to be stuck.** | **Line 429** |  |
| *assertThrow* | Invalid tag replacement. | **Line 444** |  |
| *assertThrow* | Replacement already known. | **Line 449** |  |
| *assertThrow* | Invalid tag position. | **Line 631** |  |
| *assertThrow* | Foreign tag. | **Line 632** |  |
| *assertThrow* | Tag too large. | **Line 633** |  |
| *assertThrow* | Duplicate tag position. | **Line 634** |  |
| *assertThrow* | Invalid sub id | **Line 664** |  |
| *assertThrow* | Reference to non-existing tag. | **Line 669** |  |
| *assertThrow* | Reference to tag without position. | **Line 671** |  |
| *assertThrow* | Tag too large for reserved space. | **Line 672** |  |

***AssemblyItem.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| **ErrorType** | **Message** | **Source** | **Code Snippet Example** |
| **Static\_assert** | size\_t must be at most 64-bits wide |  |  |
| **assertThrow** | Tag already has subassembly set. |  |  |
|  | **Invalid instruction.** |  |  |
| **assertThrow** | Declaration of sub-assembly tag. | **Line 190** |  |
| **assertThrow** | Invalid assembly item. | **Line 212** |  |

***CommonSubExpressionEliminator.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| **ErrorType** | **Message** | **Source** | **Code Snippet Example** |
| *assertThrow* | Incorrect final stack height. | *Line 212* |  |
| *ItemNotAvailableException()* | Undefined item requested but not available. | *Line 229* |  |
| *assertThrow* | Element already removed but still needed. | *Line 319* |  |
| *assertThrow* | Sequence constrained operation requested out of sequence. | *Line 327* |  |
| *assertThrow* | Non-generated expression without item. | *Line 329* |  |
| *assertThrow* | Undefined item requested but not available. | *Line 333* |  |
| *assertThrow* | Opcodes with more than two arguments not implemented yet. | *Line 390* |  |
| *assertThrow* | Expected arguments not present. | *Line 393* |  |
| *assertThrow* | Invalid number of return values. | *Line 413* |  |
| *assertThrow* | Element requested but is not present. | *Line 424* |  |
| *assertThrow* | Stack too deep, try removing local variables. | *Line 472*  *Line 485* |  |
| *assertThrow* | Invalid stack access. | *Line 473*  *Line 486* |  |

***ConstantOptimiser.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| **ErrorType** | **Message** | **Source** | **Code Snippet Example** |
| *assertThrow* | Empty bytecode generated. | *Line 98* |  |
| *assertThrow* | Invalid number encoding. | *Line 154* |  |
| *assertThrow* | Shift generated for invalid EVM version. | *Line 289*  *Line 291* |  |
| *assertThrow* | Invalid shift generated. | *Line 298*  *Line 300* |  |

***ControlFlowGraph.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| **ErrorType** | **Message** | **Source** | **Code Snippet Example** |
| *assertThrow* | **Tag number too large.** | **Line 40** |  |
| *assertThrow* | **Successor block not found.** | **Line 128** |  |
| *assertThrow* | **Successor already has predecessor.** | **Line 177** |  |
| *assertThrow* | **Last pushed tag not at end of pushed list.** | **Line 210** |  |
| *assertThrow* | **Out of block IDs.** | **Line 371** |  |

***ExpressionClasses.cpp***

Rule list not properly initialized.

(assertThrow, line 187)

***GasMeter.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| **ErrorType** | **Message** | **Source** | **Code Snippet Example** |
| *assertThrow* | Invalid gas tier for instruction | **Line 259** |  |
| *assertThrow* | Gas cost exceeds 256 bits. | **Line 273** |  |

***Instruction.cpp***

"<INVALID\_INSTRUCTION: " + toString((unsigned)\_inst) + ">"

(InstructionInfo, line 375)

***JumpdestRemover.cpp***

Sub-assembly tag used as label.

assertThrow, line 46

***KnownState.cpp***

Swap on same stack elements.

assertThrow, line 280

***PeepholeOptimiser.cpp***

Peephole optimizer failed to apply identity.

assertThrow, line 336

***SimplificationRules.cpp***

|  |  |  |  |
| --- | --- | --- | --- |
| **ErrorType** | **Message** | **Source** | **Code Snippet Example** |
| *assertThrow* | Rule list not properly initialized. | **Line 95** |  |

**Files from** */libyul*

* */backends*
* */optimiser*
* *AsmAnalysis.cpp*
* *AsmParser.cpp*

**Files from** */solc*

|  |  |  |  |
| --- | --- | --- | --- |
| **Status** | **Header files** | **Status** | Standard files |
|  | [CommandLineInterface.h](https://github.com/ethereum/solidity/blob/develop/solc/CommandLineInterface.h) |  | [CommandLineInterface.cpp](https://github.com/ethereum/solidity/blob/develop/solc/CommandLineInterface.cpp) |
|  | [main.cpp](https://github.com/ethereum/solidity/blob/develop/solc/main.cpp) |  |  |