

By OCamlPro

September 7, 2021

Table of Major and Critical Issues

Critical issue:	Constructor for	Data (fake)				28
Critical issue:	Constructor for	${\bf Index\ (fake)\ .\ .\ .\ .}$			•	3
Critical issue:	Constructor for	IndexBasis (fake)				38
Critical issue:	Constructor for	Manager (fake)				43
Critical issue:	Constructor for	NftRoot (fake)				49

Contents

1	Intr	roduction	7
2	Ove	erview	8
3	Lib	rary Modules	9
	3.1	Module "Constants.sol"	0
		3.1.1 Pragmas	0
		3.1.2 Contract Definitions	0
	3.2	Module "Errors.sol"	1
		3.2.1 Pragmas	1
		3.2.2 Contract Definitions	1
	3.3	Module "true_nft_audit.sol"	2
		3.3.1 Imports	2
4	$Int \epsilon$	erface Modules 1	3
	4.1	Module "IData.sol"	4
		4.1.1 Pragmas	4
		4.1.2 Contract Definitions	4
	4.2	Module "IIndex.sol"	5
		4.2.1 Pragmas	5
		4.2.2 Contract Definitions	5
	4.3	Module "IIndexBasis.sol"	6
		4.3.1 Pragmas	6
		4.3.2 Contract Definitions	6
5	Cor	ntract Modules 1	7
	5.1	Module "Data.sol"	8
		5.1.1 Pragmas	8
		5.1.2 Imports	8
			8
	5.2	Module "DataResolver.sol"	9
		5.2.1 Pragmas	9
			9
			9

3 CONTENTS

	5.3	Module "Index.sol"
		5.3.1 Pragmas
		5.3.2 Imports
		5.3.3 Contract Definitions
	5.4	Module "IndexBasis.sol"
		5.4.1 Pragmas
		5.4.2 Imports
		5.4.3 Contract Definitions
	5.5	Module "IndexResolver.sol"
		5.5.1 Pragmas
		5.5.2 Imports
		5.5.3 Contract Definitions
	5.6	Module "Manager.sol"
	0.0	5.6.1 Pragmas
		5.6.2 Imports
		5.6.3 Contract Definitions
	5.7	Module "NftRoot.sol"
	J.,	5.7.1 Pragmas
		5.7.2 Imports
		5.7.3 Contract Definitions
6	Con	tract Data 25
_	6.1	Overview
	6.2	Contract Inheritance
	6.3	Static Variable Definitions
	6.4	Variable Definitions
	6.5	Constructor Definitions
	0.0	6.5.1 Constructor
	6.6	Public Method Definitions
	0.0	6.6.1 Function destruct
		6.6.2 Function getInfo
		6.6.3 Function getOwner
	6.7	Internal Method Definitions
	0.1	6.7.1 Function deployIndex
		0.7.1 Tunetion deploymack
7	Con	tract DataResolver 31
	7.1	Overview
	7.2	Variable Definitions
	7.3	Public Method Definitions
		7.3.1 Function resolveCodeHashData
		7.3.2 Function resolveData
	7.4	Internal Method Definitions
	•	7.4.1 Function _buildDataCode
		7.4.2 Function _buildDataState

CONTENTS 3

4 CONTENTS

8	Con	tract Index	34
	8.1	Overview	34
	8.2	Contract Inheritance	34
	8.3	Static Variable Definitions	34
	8.4	Variable Definitions	35
	8.5	Constructor Definitions	35
		8.5.1 Constructor	35
	8.6	Public Method Definitions	36
		8.6.1 Function destruct	36
		8.6.2 Function getInfo	36
9	Con	tract IndexBasis	37
	9.1	Overview	37
	9.2	Static Variable Definitions	37
	9.3	Modifier Definitions	38
		9.3.1 Modifier onlyRoot	38
	9.4	Constructor Definitions	38
		9.4.1 Constructor	38
	9.5	Public Method Definitions	38
		9.5.1 Function destruct	38
		9.5.2 Function getInfo \dots	38
10	Con	tract IndexResolver	39
	10.1	Overview	39
	10.2	Variable Definitions	39
	10.3	Public Method Definitions	40
		10.3.1 Function resolveCodeHashIndex	40
		10.3.2 Function resolveIndex	40
	10.4	Internal Method Definitions	40
		10.4.1 Function _buildIndexCode	40
		10.4.2 Function _buildIndexState	40
11	Con	tract Manager	42
	11.1	Overview	42
	11.2	Variable Definitions	42
		Constructor Definitions	43
		11.3.1 Constructor	43
	11.4	Public Method Definitions	43
		11.4.1 Function deployRoot	43
	11.5	Internal Method Definitions	43
		11.5.1 Function _buildNftRootState	43

CONTENTS 4

5 CONTENTS

12	Con	tract NftRoot	45
	12.1	Overview	45
	12.2	Contract Inheritance	45
	12.3	Static Variable Definitions	46
	12.4	Variable Definitions	48
	12.5	Modifier Definitions	49
		12.5.1 Modifier onlyOwner	49
	12.6	Constructor Definitions	49
		12.6.1 Constructor	49
	12.7	Public Method Definitions	50
		12.7.1 Function burn	50
		12.7.2 Function deployBasis	50
		12.7.3 Function destructBasis	51
		12.7.4 Function getInfo	51
		12.7.5 Function mintNft	51
		12.7.6 Function setPrice	52

CONTENTS 5

To edit this document

In the report.tex file, choose:

- \soldraftfalse to remove draft mode (watermarks, advises)
- \solmodulestrue to display modules by chapter instead of contracts
- \bullet \soltable strue to display tables for parameters and returns
- \solissuesfalse to remove the table of issues

Issues can be entered with:

- $\bullet \ \backslash issueCritical\{title\}\{text\}$
- $\sin {title}{text}$
- $\bullet \ \backslash issueMinor\{title\}\{text\}$

Introduction

Overview

Library Modules

3.1 Module "Constants.sol"

3.1.1 Pragmas

ton	-solidity $>= 0.43.0$	

3.1.2 Contract Definitions

• Constants

3.2 Module "Errors.sol"

3.2.1 Pragmas

ton	-solidity $>= 0.43.0$	

3.2.2 Contract Definitions

• Errors

$3.3 \quad Module \ "true_nft_audit.sol"$

3.3.1 Imports

/share/surfer/src/NftRoot.sol	
/share/surfer/src/Manager.sol	

Interface Modules

4.1Module "IData.sol"

4.1.1 Pragmas

ton -solidity $>= 0.43.0$

4.1.2 Contract Definitions

• IData

4.2 Module "IIndex.sol"

4.2.1 Pragmas

4.2.2 Contract Definitions

• IIndex

4.3 Module "IIndexBasis.sol"

4.3.1 Pragmas

ton	-solidity $>= 0.43.0$	

4.3.2 Contract Definitions

• IIndexBasis

Contract Modules

5.1 Module "Data.sol"

5.1.1 Pragmas

ton	-solidity $>=0.43.0$	
AbiHeader	expire	
AbiHeader	time	

5.1.2 Imports

./resolvers/IndexResolver.sol	
./interfaces/IData.sol	
./libraries/Constants.sol	
./libraries/Errors.sol	

5.1.3 Contract Definitions

• Data

5.2 Module "DataResolver.sol"

5.2.1 Pragmas

ton	-solidity $>= 0.43.0$	
AbiHeader	expire	
AbiHeader	time	

5.2.2 Imports

../Data.sol

5.2.3 Contract Definitions

 \bullet DataResolver

5.3 Module "Index.sol"

5.3.1 Pragmas

ton	-solidity $>=0.43.0$	
AbiHeader	expire	
AbiHeader	time	

5.3.2 Imports

./interfaces/IIndex.sol	
./libraries/Errors.sol	

5.3.3 Contract Definitions

 \bullet Index

5.4 Module "IndexBasis.sol"

5.4.1 Pragmas

ton	-solidity $>=0.43.0$	
AbiHeader	expire	
AbiHeader	time	

5.4.2 Imports

./libraries/Errors.sol

5.4.3 Contract Definitions

 \bullet IndexBasis

5.5 Module "IndexResolver.sol"

5.5.1 Pragmas

ton	-solidity $>= 0.43.0$	
AbiHeader	expire	
AbiHeader	time	

5.5.2 Imports

../Index.sol

5.5.3 Contract Definitions

 \bullet IndexResolver

5.6 Module "Manager.sol"

5.6.1 Pragmas

ton	-solidity $>= 0.43.0$	
AbiHeader	expire	
AbiHeader	time	

5.6.2 Imports

./NftRoot.sol	
./libraries/Constants.sol	
./libraries/Errors.sol	

5.6.3 Contract Definitions

• Manager

5.7 Module "NftRoot.sol"

5.7.1 Pragmas

ton	-solidity $>=0.43.0$	
AbiHeader	expire	
AbiHeader	time	

5.7.2 Imports

./resolvers/IndexResolver.sol	
./resolvers/DataResolver.sol	
./IndexBasis.sol	
./interfaces/IIndexBasis.sol	
./libraries/Constants.sol	
./libraries/Errors.sol	

5.7.3 Contract Definitions

 \bullet NftRoot

Contract Data

Contents	
6.1	Overview
6.2	Contract Inheritance
6.3	Static Variable Definitions 25
6.4	Variable Definitions
6.5	Constructor Definitions
	6.5.1 Constructor
6.6	Public Method Definitions 29
	6.6.1 Function destruct
	6.6.2 Function getInfo
	6.6.3 Function getOwner
6.7	Internal Method Definitions 30
	6.7.1 Function deployIndex
	- v

6.1 Overview

In file Data.sol

6.2 Contract Inheritance

IData	
IndexResolver	

6.3 Static Variable Definitions

|--|

17 uint256 static _id;

6.4 Variable Definitions

address	_addrRoot	
		used in @9.Data.destruct
		used in @9.Data.destruct
		used in @9.Data.deployIndex
		used in @9.Data.deployIndex
		used in @9.Data.deployIndex
		assigned in @9.Data.:constructor
		used in @9.Data.:constructor
address	_addrOwner	
		used in @9.Data.getOwner
		used in @9.Data.destruct
		used in @9.Data.destruct
		assigned in @9.Data.:constructor
		used in @9.Data.:constructor
address	_addrAuthor	
		used in @9.Data.getInfo
		assigned in @9.Data.:constructor
		used in @9.Data.:constructor
string	_name	
		used in @9.Data.getInfo
		assigned in @9.Data.:constructor
		used in @9.Data.:constructor
string	_description	
	1	used in @9.Data.getInfo
		assigned in @9.Data.:constructor
		used in @9.Data.:constructor
string	_tokenCode	
		used in @9.Data.getInfo
		assigned in @9.Data.:constructor
		used in @9.Data.:constructor
uint64	_creationDate	
		used in @9.Data.getInfo
		assigned in @9.Data.:constructor
		used in @9.Data.:constructor
string	_comment	
		used in @9.Data.getInfo
		assigned in @9.Data.:constructor
		used in @9.Data.:constructor
mapping (uint128 $=>$ bytes)	_content	
		used in @9.Data.getInfo
		assigned in @9.Data.:constructor
		used in @9.Data.:constructor

```
14
       address _addrRoot;
15
       address _addrOwner;
       address _addrAuthor;
16
19
       string _name;
20
       string _description;
21
       string _tokenCode;
       uint64 _creationDate;
22
       string _comment;
23
26
       mapping(uint128 => bytes) _content;
```

6.5 Constructor Definitions

6.5.1 Constructor

Critical issue: Constructor for Data (fake)

loren ipsum loren

loren ipsum loren

```
28
       constructor(
29
           address addrOwner,
30
            TvmCell codeIndex,
31
           address addrAuthor,
32
            string name,
33
            string description,
34
           string tokenCode,
35
           uint64 creationDate,
36
           string comment,
37
           uint128 index,
38
           bytes part
39
       ) public {
40
            optional(TvmCell) optSalt = tvm.codeSalt(tvm.code());
41
            require(optSalt.hasValue(), Errors.ERROR_EMPTY_SALT);
42
            (address addrRoot) = optSalt
43
                .get()
44
                .toSlice()
45
                .decode(address);
46
            require(msg.sender == addrRoot, Errors.
                ERROR_MESSAGE_SENDER_IS_NOT_ROOT);
```

```
47
           require(msg.value >= Constants.MIN_FOR_DEPLOY);
48
            _addrRoot = addrRoot;
            _addrOwner = addrOwner;
49
50
           _addrAuthor = addrAuthor;
51
           _name = name;
52
            _description = description;
            _tokenCode = tokenCode;
53
54
            _creationDate = creationDate;
55
           _comment = comment;
56
            _codeIndex = codeIndex;
57
58
            _content[index] = part;
59
60
            deployIndex(addrOwner);
61
```

6.6 Public Method Definitions

6.6.1 Function destruct

• TODO

```
77
       function destruct(address recipient) public {
           require(msg.sender == _addrRoot, Errors.
78
               ERROR_MESSAGE_SENDER_IS_NOT_ROOT);
79
80
           address oldIndexOwner = resolveIndex(address(0), address(
               this), _addrOwner);
81
           IIndex(oldIndexOwner).destruct();
           address oldIndexOwnerRoot = resolveIndex(_addrRoot, address
82
               (this), _addrOwner);
83
           IIndex(oldIndexOwnerRoot).destruct();
84
85
           recipient.transfer(0, false, 64);
86
           selfdestruct(recipient);
```

6.6.2 Function getInfo

```
94
      function getInfo() public view override
            mapping(uint128 => bytes) content,
96
97
            address author,
98
            string name,
99
            string description,
            string tokenCode,
100
101
            uint64 creationDate,
102
            string comment
103
        ) {
            content = _content;
104
```

```
author = _addrAuthor;
name = _name;
description = _description;
tokenCode = _tokenCode;
creationDate = _creationDate;
comment = _comment;
}
```

6.6.3 Function getOwner

• TODO

```
89     function getOwner() public view override returns(address
          addrOwner, address addrNftData) {
90         addrOwner = _addrOwner;
91         addrNftData = address(this);
92    }
```

6.7 Internal Method Definitions

6.7.1 Function deployIndex

```
63
       function deployIndex(address owner) private {
            TvmCell codeIndexOwner = _buildIndexCode(address(0), owner)
64
65
            TvmCell stateIndexOwner = _buildIndexState(codeIndexOwner,
                address(this));
66
            new Index
                {stateInit: stateIndexOwner, value: Constants.
67
                    DEPLOY_INDEX_FEE, flag: 0}
                (_addrRoot);
68
69
70
            TvmCell codeIndexOwnerRoot = _buildIndexCode(_addrRoot,
                owner);
71
            TvmCell stateIndexOwnerRoot = _buildIndexState(
                codeIndexOwnerRoot, address(this));
72
            new Index
73
                \{ \verb|stateInit|: \verb|stateIndexOwnerRoot|, \verb|value|: Constants|.
                    DEPLOY_INDEX_FEE, flag: 0}
74
                (_addrRoot);
```

Contract DataResolver

Contents				
7.1	Ove	rview		
7.2	Vari	Variable Definitions		
7.3	Pub	lic Method Definitions		
	7.3.1	Function resolveCodeHashData		
	7.3.2	Function resolveData		
7.4	Inte	rnal Method Definitions		
	7.4.1	Function _buildDataCode		
	7.4.2	Function _buildDataState		

7.1 Overview

In file DataResolver.sol

7.2 Variable Definitions

TvmCell	_codeData			
		assigned	in	@2.Nft-
		Root.:consti	ructor	
		used in @2.1	NftRoo	t.:constructor
		used	in	@7.DataRe-
		solverbuild	lDataC	ode

11 TvmCell _codeData;

7.3 Public Method Definitions

7.3.1 Function resolveCodeHashData

• TODO

7.3.2 Function resolveData

• TODO

```
function resolveData(
    address addrRoot,
    uint256 id

public view returns (address addrData) {
    TvmCell code = _buildDataCode(addrRoot);
    TvmCell state = _buildDataState(code, id);
    uint256 hashState = tvm.hash(state);
    addrData = address.makeAddrStd(0, hashState);
}
```

7.4 Internal Method Definitions

7.4.1 Function _buildDataCode

• TODO

```
function _buildDataCode(address addrRoot) internal virtual view
    returns (TvmCell) {

TvmBuilder salt;

salt.store(addrRoot);

return tvm.setCodeSalt(_codeData, salt.toCell());

}
```

7.4.2 Function _buildDataState

Contract Index

Contents	
8.1	Overview
8.2	Contract Inheritance
8.3	Static Variable Definitions
8.4	Variable Definitions
8.5	Constructor Definitions
	8.5.1 Constructor
8.6	Public Method Definitions
	8.6.1 Function destruct
	8.6.2 Function getInfo

8.1 Overview

In file Index.sol

8.2 Contract Inheritance

IIndex				
--------	--	--	--	--

8.3 Static Variable Definitions

address	_addrData	
		used in @10.Index.getInfo
		used in @10.Index.destruct
		used in @10.Index.destruct
		used in @10.Index.:constructor

```
13 address static _addrData;
```

8.4 Variable Definitions

address	_addrRoot	
		used in @10.Index.getInfo
		assigned in @10.In-
		dex.:constructor
		used in @10.Index.:constructor
		assigned in @10.In-
		dex.:constructor
		used in @10.Index.:constructor
address	_addrOwner	
		used in @10.Index.getInfo
		assigned in @10.In-
		dex.:constructor
		used in @10.Index.:constructor

```
11   address _addrRoot;
12   address _addrOwner;
```

8.5 Constructor Definitions

8.5.1 Constructor

Critical issue: Constructor for Index (fake)

loren ipsum loren

loren ipsum loren

```
15
         constructor(address root) public {
             optional(TvmCell) optSalt = tvm.codeSalt(tvm.code());
require(optSalt.hasValue(), Errors.ERROR_EMPTY_SALT);
16
17
18
              (address addrRoot, address addrOwner) = optSalt
19
                   .get()
20
                   .toSlice()
21
                   .decode(address, address);
22
              require(msg.sender == _addrData, Errors.
                  ERROR_MESSAGE_SENDER_IS_NOT_OWNER);
23
             tvm.accept();
24
              _addrRoot = addrRoot;
```

8.6 Public Method Definitions

8.6.1 Function destruct

 \bullet TODO

8.6.2 Function getInfo

 \bullet TODO

```
function getInfo() public view override returns (
    address addrRoot,
    address addrOwner,
    address addrData

) {
    addrRoot = _addrRoot;
    addrOwner = _addrOwner;
    addrData = _addrData;
}
```

Contract IndexBasis

Contents		
9.1	Ove	rview
9.2	Stat	ic Variable Definitions
9.3	Mod	ifier Definitions
	9.3.1	Modifier onlyRoot
9.4	Cons	structor Definitions
	9.4.1	Constructor
9.5	Pub	lic Method Definitions
	9.5.1	Function destruct
	9.5.2	Function getInfo

9.1 Overview

In file IndexBasis.sol

9.2 Static Variable Definitions

address	_addrRoot	
		used in @5.IndexBasis.getInfo
		used in @5.IndexBasis.destruct
uint256	_codeHashData	
		used in @5.IndexBasis.getInfo

```
9 address static _addrRoot;
10 uint256 static _codeHashData;
```

9.3 Modifier Definitions

9.3.1 Modifier onlyRoot

9.4 Constructor Definitions

9.4.1 Constructor

Critical issue: Constructor for IndexBasis (fake)

loren ipsum loren

loren ipsum loren

• TODO

```
18 constructor() public onlyRoot {}
```

9.5 Public Method Definitions

9.5.1 Function destruct

• TODO

```
25    function destruct() public onlyRoot {
26         selfdestruct(_addrRoot);
27    }
```

9.5.2 Function getInfo

Contract IndexResolver

Contents 10.1 Overview 39 10.2 Variable Definitions 39 10.3 Public Method Definitions 40 10.3.1 Function resolveCodeHashIndex 40 10.3.2 Function resolveIndex 40 10.4 Internal Method Definitions 40 10.4.1 Function _buildIndexCode 40 10.4.2 Function _buildIndexState 40

10.1 Overview

In file IndexResolver.sol

10.2 Variable Definitions

TvmCell	_codeIndex			
		used in @2.NftRoot.mintNft		
		assigned in @2.Nft-		
		Root.:constructor		
		used in @2.NftRoot.:constructor		
		assigned in @9.Data.:constructor		
		used in @9.Data.:constructor		
		used in @8.IndexRe-		
		solverbuildIndexCode		

11 TvmCell _codeIndex;

10.3 Public Method Definitions

10.3.1 Function resolveCodeHashIndex

• TODO

```
function resolveCodeHashIndex(
    address addrRoot,
    address addrOwner

public view returns (uint256 codeHashIndex) {
    return tvm.hash(_buildIndexCode(addrRoot, addrOwner));
}
```

10.3.2 Function resolveIndex

• TODO

```
function resolveIndex(
21
          address addrRoot,
22
           address addrData,
23
           address addr0wner
24
      ) public view returns (address addrIndex) {
25
           TvmCell code = _buildIndexCode(addrRoot, addrOwner);
26
           TvmCell state = _buildIndexState(code, addrData);
27
           uint256 hashState = tvm.hash(state);
           addrIndex = address.makeAddrStd(0, hashState);
28
```

10.4 Internal Method Definitions

10.4.1 Function _buildIndexCode

TODO

```
function _buildIndexCode(
   address addrRoot,
   address addrOwner

internal virtual view returns (TvmCell) {
   TvmBuilder salt;
   salt.store(addrRoot);
   salt.store(addrOwner);
   return tvm.setCodeSalt(_codeIndex, salt.toCell());
}
```

10.4.2 Function _buildIndexState

```
function _buildIndexState(
    TvmCell code,
    address addrData

internal virtual pure returns (TvmCell) {
    return tvm.buildStateInit({
        contr: Index,
        varInit: {_addrData: addrData},
        code: code
}

});
```

Contract Manager

Contents

Contents	
11.1 Overview	
11.2 Variable Definitions	
11.3 Constructor Definitions	
11.3.1 Constructor	
11.4 Public Method Definitions 43	
11.4.1 Function deployRoot	
11.5 Internal Method Definitions 43	
11.5.1 Function _buildNftRootState	

11.1 Overview

In file Manager.sol

11.2 Variable Definitions

TvmCell	_rootCode			
		used	in	@1.Man-
		agerbuil	dNftRootS	State
		assigned	in	@1.Man-
		ager.:constructor		
		used in @1.Manager.:constructor		

13 TvmCell _rootCode;

11.3 Constructor Definitions

11.3.1 Constructor

Critical issue: Constructor for Manager (fake)

loren ipsum loren

loren ipsum loren

• TODO

11.4 Public Method Definitions

11.4.1 Function deployRoot

• TODO

```
20
        function deployRoot(
21
             address addrOwner,
22
             TvmCell codeIndex,
23
             TvmCell codeData,
24
             string name,
25
             string description,
26
             string tokenCode,
27
             uint256 totalSupply,
             uint128 index,
28
29
             bytes part
        ) public view {
30
31
             tvm.accept();
32
             TvmCell stateNftRoot = _buildNftRootState(addrOwner);
new NftRoot {stateInit: stateNftRoot, value: Constants.
33
34
                  DEPLOY_INDEX_FEE}( codeIndex, codeData, name,
                  description, tokenCode, totalSupply, index, part);
35
```

11.5 Internal Method Definitions

11.5.1 Function _buildNftRootState

Contract NftRoot

Contents	
12.1 Overview	5
12.2 Contract Inheritance 45	5
12.3 Static Variable Definitions 46	3
12.4 Variable Definitions 48	3
12.5 Modifier Definitions 49	9
12.5.1 Modifier onlyOwner	9
12.6 Constructor Definitions 49	9
12.6.1 Constructor	9
12.7 Public Method Definitions)
12.7.1 Function burn	Э
12.7.2 Function deployBasis	О
12.7.3 Function destructBasis 5	1
12.7.4 Function getInfo	1
12.7.5 Function mintNft	1
12.7.6 Function setPrice	2

12.1 Overview

In file NftRoot.sol

12.2 Contract Inheritance

DataResolver	
IndexResolver	

12.3 Static Variable Definitions

	address	_addrOwne	r	
32	addre	ess static	addrOwner:	

12.4 Variable Definitions

uint256	_totalMinted	
		assigned in @2.NftRoot.mintNft
		used in @2.NftRoot.mintNft
		used in @2.NftRoot.mintNft
		used in @2.NftRoot.mintNft
address	_addrBasis	
		used in @2.Nft-
		Root.destructBasis
		assigned in @2.Nft-
		Root.deployBasis
		used in @2.NftRoot.deployBasis
uint256	_totalSupply	
		used in @2.NftRoot.mintNft
		used in @2.NftRoot.getInfo
		assigned in @2.Nft-
		Root.:constructor
		used in @2.NftRoot.:constructor
string	_name	
		used in @2.NftRoot.mintNft
		used in @2.NftRoot.getInfo
		assigned in @2.Nft-
		Root.:constructor
		used in @2.NftRoot.:constructor
string	_description	
	1	used in @2.NftRoot.mintNft
		used in @2.NftRoot.getInfo
		assigned in @2.Nft-
		Root.:constructor
		used in @2.NftRoot.:constructor
string	_tokenCode	
		used in @2.NftRoot.mintNft
		used in @2.NftRoot.getInfo
		assigned in @2.Nft-
		Root.:constructor
		used in @2.NftRoot.:constructor
mapping (uint128 $=>$ bytes)	_content	
		used in @2.NftRoot.mintNft
		used in @2.NftRoot.getInfo
		assigned in @2.Nft-
		Root.:constructor
		used in @2.NftRoot.:constructor
uint128	_price	
		assigned in @2.NftRoot.setPrice
		used in @2.NftRoot.setPrice
CHAPTER 12. CONTRACT	NFTROOT	used in @2.NftRoot.getInfo 48
		used in @2.NftRoot.burn
		assigned in @2.Nft-
		Root.:constructor
		used in @2.NftRoot.:constructor
	1	I .

```
uint256 _totalMinted;
18
19
       address _addrBasis;
21
       uint256 _totalSupply;
23
       string _name;
24
       string _description;
25
       string _tokenCode;
28
       mapping(uint128 => bytes) _content;
30
       uint128 _price;
```

12.5 Modifier Definitions

12.5.1 Modifier onlyOwner

12.6 Constructor Definitions

12.6.1 Constructor

Critical issue: Constructor for NftRoot (fake)

loren ipsum loren

loren ipsum loren

```
40
        constructor(
41
            TvmCell codeIndex,
42
            TvmCell codeData,
43
            string name,
44
            string description,
45
            string tokenCode,
            uint256 totalSupply,
46
47
            uint128 index,
            bytes part
48
49
        ) public {
```

```
50
            tvm.accept();
51
            _codeIndex = codeIndex;
            _codeData = codeData;
52
53
            _name = name;
            _description = description;
54
55
            _tokenCode = tokenCode;
56
            _totalSupply = totalSupply;
57
58
            _content[index] = part;
59
60
            _price = 1 ton;
61
```

12.7 Public Method Definitions

12.7.1 Function burn

• TODO

12.7.2 Function deployBasis

```
function deployBasis(TvmCell codeIndexBasis) public onlyOwner {
85
           require(msg.value > 0.5 ton, Errors.ERROR_NOT_ENOUGH_GRAMS)
86
87
            uint256 codeHasData = resolveCodeHashData();
            TvmCell state = tvm.buildStateInit({
88
89
                contr: IndexBasis,
90
                varInit: {
91
                    _codeHashData: codeHasData,
92
                    _addrRoot: address(this)
93
94
                code: codeIndexBasis
95
           });
96
            _addrBasis = new IndexBasis{stateInit: state, value: 0.4
               ton}();
```

12.7.3 Function destructBasis

• TODO

```
99 function destructBasis() public view onlyOwner {
100 IIndexBasis(_addrBasis).destruct();
101 }
```

12.7.4 Function getInfo

• TODO

```
103
         function getInfo() public view returns (
104
             mapping(uint128 => bytes) content,
105
             string name,
             string description,
106
107
             string tokenCode,
108
             uint256 totalSupply,
109
             uint128 price
110
111
             content = _content;
             name = _name;
112
113
             description = _description;
114
             tokenCode = _tokenCode;
115
             totalSupply = _totalSupply;
116
             price = _price;
```

12.7.5 Function mintNft

```
function mintNft(uint64 creationDate, string comment, address
63
            owner) public onlyOwner {
            require(msg.value >= 1.6 ton, Errors.ERROR_NOT_ENOUGH_GRAMS
64
                );
            require(_totalMinted <= _totalSupply, Errors.</pre>
65
                ERROR_MINTED_TOO_MUCH);
66
            TvmCell codeData = _buildDataCode(address(this));
            TvmCell stateData = _buildDataState(codeData, _totalMinted)
67
68
            new Data
69
                {stateInit: stateData, value: 1.5 ton} (
70
                    owner,
71
                    _codeIndex,
72
                    msg.sender,
73
                    _name,
74
                     _description,
75
                     _tokenCode,
76
                    creationDate,
77
                    comment,
78
                    0,
79
                     _content[0]
```

```
80 );
81 82 __totalMinted++;
83 }
```

12.7.6 Function setPrice

```
function setPrice(uint128 price) public onlyOwner {
    _price = price;
}
```