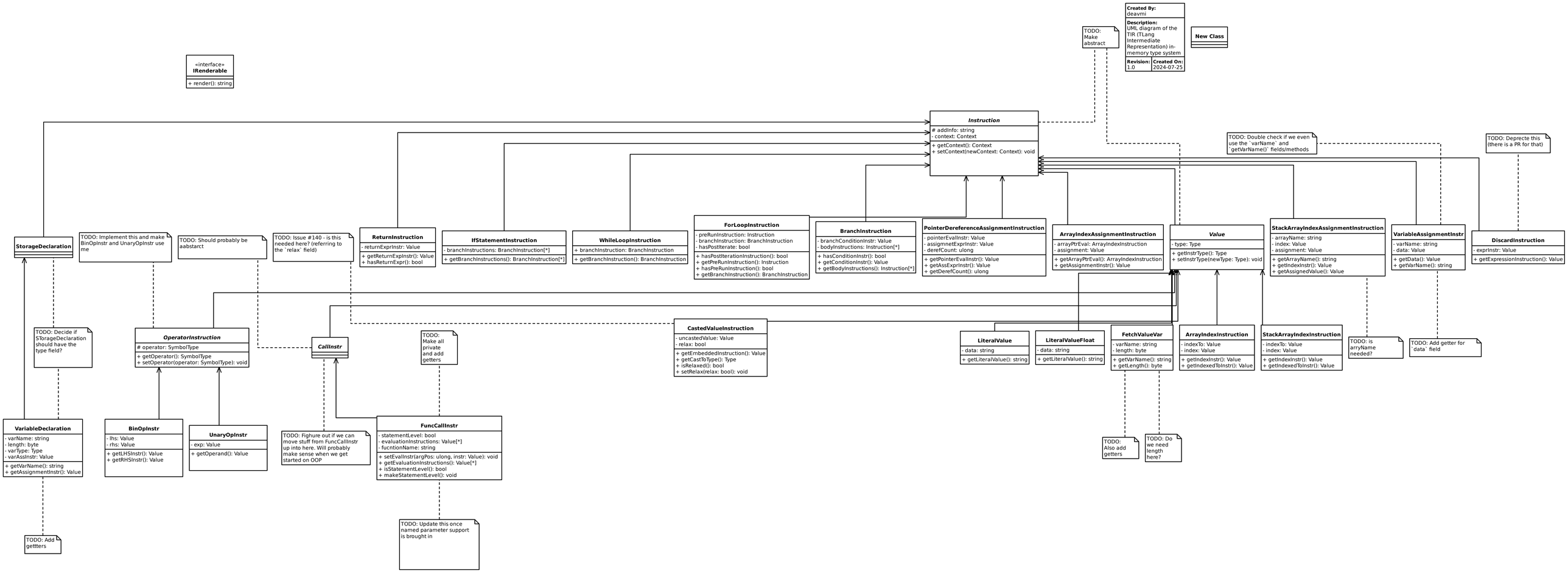
«interface»
IRenderable
+ render(): string
Created By:
deavmi
Description:
UML diagram of the TIR (TLang Intermediate Representation) in-memory type system
Revision:
1.0
Created On:
2024-07-25
New Class
TODO:
Make
abstract
Instruction
addInfo: string
- context: Context
+ getContext(): Context
+ setContext(newContext: Context): void
TODO: Double check if we even use the `varName` and `getVarName()` fields/methods
TODO: Deprecte this (there is a PR for that)
StorageDeclaration
TODO: Implement this and make BinOpInstr and UnaryOpInstr use me
TODO: Should probably be aabstarct
TODO: Issue #140 - is this needed here? (referring to the `relax` field)
ReturnInstruction
- returnExprInstr: Value
+ getReturnExpInstr(): Value
+ hasReturnExpr(): bool
IfStatementInstruction
- branchInstructions: BranchInstruction[*]
+ getBranchInstructions(): BranchInstruction[*]
WhileLoopInstruction
+ branchInstruction: BranchInstruction
+ getBranchInstruction(): BranchInstruction
ForLoopInstruction
- preRunInstruction: Instruction
- branchInstruction: BranchInstruction
- hasPostIterate: bool
+ hasPostIterationInstruction(): bool
+ getPreRunInstruction(): Instruction
+ hasPreRunInstruction(): bool
+ getBranchInstruction(): BranchInstruction
BranchInstruction
- branchConditionInstr: Value
- bodyInstructions: Instruction[*]
+ hasConditionInstr(): bool
+ getConditionInstr(): Value
+ getBodyInstructions(): Instruction[*]
PointerDereferenceAssignmentInstruction
- pointerEvalInstr: Value
- assignmnetExprInstr: Value
- derefCount: ulong
+ getPointerEvalInstr(): Value
+ getAssExprInstr(): Value
+ getDerefCount(): ulong
ArrayIndexAssignmentInstruction
- arrayPtrEval: ArrayIndexInstruction
- assignment: Value
+ getArrayPtrEval(): ArrayIndexInstruction
+ getAssignmentInstr(): Value
Value
- type: Type
+ getInstrType(): Type
+ setInstrType(newType: Type): void
StackArrayIndexAssignmentInstruction
- arrayName: string
- index: Value
- assignment: Value
+ getArrayName(): string
+ getIndexInstr(): Value
+ getAssignedValue(): Value
VariableAssignmentInstr
- varName: string
- data: Value
+ getData(): Value
+ getVarName(): string
DiscardInstruction
- exprInstr: Value
+ getExpressionInstruction(): Value
TODO: Decide if SToragaDeclaration should have the type field?
OperatorInstruction
operator: SymbolType
+ getOperator(): SymbolType
+ setOperator(operator: SymbolType): void
CallInstr
TODO:
Make all
private
and add
getters
CastedValueInstruction
- uncastedValue: Value
- relax: bool
+ getEmbeddedInstruction(): Value
+ getCastToType(): Type
+ isRelaxed(): bool
+ setRelax(relax: bool): void
LiteralValue
- data: string
+ getLiteralValue(): string
LiteralValueFloat
- data: string
+ getLiteralValue(): string
FetchValueVar
- varName: string
- length: byte
+ getVarName(): string
+ getLength(): byte
ArrayIndexInstruction
- indexTo: Value
- index: Value
+ getIndexInstr(): Value
+ getIndexedToInstr(): Value
StackArrayIndexInstruction
- indexTo: Value
- index: Value
+ getIndexInstr(): Value
+ getIndexedToInstr(): Value
TODO: is arryName needed?
TODO: Add getter for `data` field
VariableDeclaration
- varName: string
- length: byte
- varType: Type
- varAssInstr: Value
+ getVarName(): string
+ getAssignmentInstr(): Value
BinOpInstr
- lhs: Value
- rhs: Value
+ getLHSInstr(): Value
+ getRHSInstr(): Value
UnaryOpInstr
- exp: Value
+ getOperand(): Value
TODO: Fighure out if we can move stuff from FuncCallInstr up into here. Will probably make sense when we get started on OOP
FuncCallInstr
- statementLevel: bool
- evaluationInstructions: Value[*]
- fucntionName: string
+ setEvalInstr(argPos: ulong, instr: Value): void
+ getEvaluationInstructions(): Value[*]
+ isStatementLevel(): bool
+ makeStatementLevel(): void
TODO:
Also add
getters
TODO: Do
we need
length
here?
TODO: Add
gettters
TODO: Update this once named parameter support is brought in