**FlowType**
<<enumeration>>
(org::jimmycollins::networktraffic::util::FlowFactory)
TCP
UDP
ICMP
ARP
RTP
PIM
IPXSPX
RTCP
IGMP
IPV6
IPV6ICMP
UDT
ESP
UNAS
RARP

**FlowFactory**
(org::jimmycollins::networktraffic::util)
+CreateFlow(type : String) : Flow

**<<Interface>> DShieldApi**
(org::jimmycollins::networktraffic::model)
+Infocon() : String
+Ip(ip : String) : DShieldIpInfo
+Port(port : String) : DShieldPortInfo
+PortHistory(port : String) : List<DShieldPortHistoryInfo>

**DShieldApiProxy**
(org::jimmycollins::networktraffic::util)
-RemoteApi : RemoteDShield
+DShieldApiProxy()
+Infocon() : String
+Ip(ip : String) : DShieldIpInfo
+Port(port : String) : DShieldPortInfo
+PortHistory(port : String) : List<DShieldPortHistoryInfo>

**RemoteDShieldApi**
(org::jimmycollins::networktraffic::util)
-ApiBaseUrl : String = "https://www.dshield.org/api/"
+Infocon() : String
+Ip(ip : String) : DShieldIpInfo
+Port(port : String) : DShieldPortInfo
+PortHistory(port : String) : ArrayList<DShieldPortHistoryInfo>

**BinetFile**
(org::jimmycollins::networktraffic::util)
-Factory : FlowFactory
-locale : Locale = new Locale("en", "US")
-resources : ResourceBundle = ResourceBundle.getBundle("ResourcesBundle", locale)
+BinetFile(file : File)
+ParseFile(stats : FlowFileStats) : FlowFileStats

**ParsableFile**
(org::jimmycollins::networktraffic::model)
-File : File
+ParsableFile(File : File)
+GetFile() : File
+ParseFile(Stats : FlowFileStats) : FlowFileStats

**NetworkTraffic**
(org::jimmycollins::networktraffic)
+start(stage : Stage) : void
+main(args : String []) : void

**DataExporter**
(org::jimmycollins::networktraffic::util)
-command : ExportCommand
+export() : void

**<<Interface>> ExportCommand**
(org::jimmycollins::networktraffic::model)
+execute() : void

**ExportHtmlCommand**
(org::jimmycollins::networktraffic::util)
-Data : DataExport
+ExportHtmlCommand(theData : DataExport)
+execute() : void

**ExportExcelCommand**
(org::jimmycollins::networktraffic::util)
-Data : DataExport
+ExportExcelCommand(theData : DataExport)
+execute() : void

**DataExport**
(org::jimmycollins::networktraffic::model)
+SourcePortData : Map<String, Integer>
+DestinationPortData : Map<String, Integer>
+SourceIpData : Map<String, Integer>
+DestinationIpData : Map<String, Integer>
+DataExport(sourcePorts : Map<String, Integer>, destinationPorts : Map<String, Integer>, sourceIps : Map<String, Integer>, destinationIps : Map<String, Integer>)

**PieChartStrategy**
(org::jimmycollins::networktraffic)
+ChartData(chartData : Map<String, Integer>, chartTitle : String) : Chart

**LineChartStrategy**
(org::jimmycollins::networktraffic)
+ChartData(chartData : Map<String, Integer>, chartTitle : String) : Chart

**<<Interface>> DisplayStrategy**
(org::jimmycollins::networktraffic)
+ChartData(chartData : Map<String, Integer>, chartTitle : String) : Chart

**BarChartStrategy**
(org::jimmycollins::networktraffic)
+ChartData(chartData : Map<String, Integer>, chartTitle : String) : Chart

**DisplayContext**
(org::jimmycollins::networktraffic)
-chartStrategy : DisplayStrategy
+createChart(chartData : Map<String, Integer>, chartTitle : String) : Chart

**DShieldIpInfo**
(org::jimmycollins::networktraffic::model)
-IP : SimpleStringProperty
-FQDN : SimpleStringProperty
-Blocked : SimpleStringProperty
-Attacks : SimpleStringProperty
-Country : SimpleStringProperty
+DShieldIpInfo(ip : String, fqdn : String, blocked : String, attacks : String, country : String)
+SetIP(IP : SimpleStringProperty) : void
+IPProperty() : SimpleStringProperty
+SetFqdn(FQDN : SimpleStringProperty) : void
+FQDNProperty() : SimpleStringProperty
+SetBlocked(Blocked : SimpleStringProperty) : void
+BlockedProperty() : SimpleStringProperty
+SetBlocked(Blocked : SimpleStringProperty) : void
+AttacksProperty() : SimpleStringProperty
+SetAttacks(Attacks : SimpleStringProperty) : void
+CountryProperty() : SimpleStringProperty
+SetCountry(Country : SimpleStringProperty) : void

**DShieldPortHistoryInfo**
(org::jimmycollins::networktraffic::model)
-Date : String
-Records : String
-Targets : String
-Sources : String
+DShieldPortHistoryInfo(date : String, records : String, targets : String, sources : String)
+getDate() : String
+setDate(Date : String) : void
+getRecords() : String
+setRecords(Records : String) : void
+getTargets() : String
+setTargets(Targets : String) : void
+getSources() : String
+setSources(Sources : String) : void

**Utility**
(org::jimmycollins::networktraffic::util)
+ParseDateTime(value : String) : LocalDateTime
+ParseInt(value : String) : int
+ParseInetAddress(value : String) : InetAddress
+GetTopElements(list : ArrayList<String>, elementCount : int) : Map<String, Integer>
+BytesToMb(bytes : long) : long
+GenerateTimestamp() : String
+GenerateDateStamp() : String
+GetFqdn(ip : String) : String
-SortMapByValue(mapToSort : Map<K, V>) : Map<K, V>

**Database**
(org::jimmycollins::networktraffic::util)
+GetInstance() : Database
+GetConnection() : Connection

**LogLevel**
<<enumeration>>
(org::jimmycollins::networktraffic::util::LogUtil)
INFO
WARN

**LogUtil**
(org::jimmycollins::networktraffic::util)
-LOGGER : Logger = Logger.getLogger("com.jimmycollins.networktraffic")
-LOGGER_INITIALIZED : Boolean = false
-handler : FileHandler
-locale : Locale = new Locale("en", "US")
-resources : ResourceBundle = ResourceBundle.getBundle("ResourcesBundle", locale)
+Initialize() : void
+Log(level : LogLevel, message : String) : void
+Log(type : AlertType, header : String, message : String) : void

**ParsingObserver**
(org::jimmycollins::networktraffic::view::MainUserInterfaceController)
-ParsingObserver(stats : FlowFileStats)
+update() : void

**Observer**
(org::jimmycollins::networktraffic::model)
+stats : FlowFileStats
+update() : void

**FlowFileStats**
(org::jimmycollins::networktraffic::model)
-Observers : Observer = new ArrayList<>()
-Flows : Flow
-ParsedFlows : int
-UnparsableFlows : int
-TotalPackets : int
+FlowFileStats()
+IncrementUnparsableFlowCounter() : void
+AddFlow(flow : Flow) : void
+AddPackets(packetCount : int) : void
+GetParsedFlows() : int
+GetUnparsableFlows() : int
+GetFlows() : ArrayList<Flow>
+GetTotalPacketCount() : int
+attach(observer : Observer) : void
+notifyAllObservers() : void

**MainUserInterfaceController**
(org::jimmycollins::networktraffic::view)
-parsedFlowsLabel : Label
-parsingErrorsLabel : Label
-totalPacketsLabel : Label
-topSourcePortsPane : Pane
-topDestinationPortsPane : Pane
-topSourceIPsPane : Pane
-topDestinationIPsPane : Pane
-saveCurrentAnalysisBtn : Button
-saveChartsBtn : Button
-securityAnalysisButton : Button
-portAnalysisButton : Button
-mainTabPane : TabPane
-securityAnalysisTab : Tab
-portAnalysisTab : Tab
-threatLevelLabel : Label
-portAnalysisPane : HBox
-sourceIpsSecurityInfoTable : TableView
-sourcePortSecurityInfoData : ObservableList<DShieldInfo> = FXCollections.observableArrayList()
-sourceIpColumn : TableColumn
-sourceIpFqdnColumn : TableColumn
-sourceIpBlockedColumn : TableColumn
-sourceIpAttacksColumn : TableColumn
-sourceIpCountryColumn : TableColumn
-destinationIpSecurityInfoTable : TableView
-destinationIPSecurityInfoData : ObservableList<DShieldIpInfo> = FXCollections.observableArrayList()
-destinationIpColumn : TableColumn
-destinationFqdnColumn : TableColumn
-destinationIpBlockedColumn : TableColumn
-destinationIpAttacksColumn : TableColumn
-destinationIpCountryColumn : TableColumn
-sourcePortSecurityInfoTable : TableView
-sourcePortSecurityInfoData : ObservableList<DShieldPortInfo> = FXCollections.observableArrayList()
-sourcePortRecordsColumn : TableColumn
-sourcePortTargetsColumn : TableColumn
-sourcePortSourcesColumn : TableColumn
-destinationPortSecurityInfoTable : TableView
-destinationPortSecurityInfoData : ObservableList<DShieldPortInfo> = FXCollections.observableArrayList()
-destinationPortColumn : TableColumn
-destinationPortRecordsColumn : TableColumn
-destinationPortTargetsColumn : TableColumn
-sourcePortData : Map<String, Integer> = new HashMap<>()
-destinationPortData : Map<String, Integer> = new HashMap<>()
-sourceIpData : Map<String, Integer> = new HashMap<>()
-destinationIpData : Map<String, Integer> = new HashMap<>()
-chartContext : DisplayContext = new DisplayContext()
-stats : FlowFileStats = new FlowFileStats()
-file : File
-locale : Locale = new Locale("en", "US")
-resources : ResourceBundle = ResourceBundle.getBundle("ResourcesBundle", locale)
-handleParseNewFile(event : ActionEvent) : void
-handleDrawGeneralCharts(event : ActionEvent) : void
-handleLoadSession(event : ActionEvent) : void
-handleSaveSession(event : ActionEvent) : void
-handleDrawBarCharts(event : ActionEvent) : void
-handleDrawLineCharts(event : ActionEvent) : void
-drawChartsInitial(strategy : DisplayStrategy) : void
-drawGeneralCharts(strategy : DisplayStrategy) : void
-handleSecurityAnalysis(event : ActionEvent) : void
-handleExportAnalysis(event : ActionEvent) : void
-handlePortAnalysis(event : ActionEvent) : void
-performPortAnalysis(event : ActionEvent) : void
+initialize(url : URL, rb : ResourceBundle) : void

**DShieldPortInfo**
(org::jimmycollins::networktraffic::model)
-Port : SimpleStringProperty
-Records : SimpleStringProperty
-Targets : SimpleStringProperty
-Sources : SimpleStringProperty
+DShieldPortInfo(port : String, records : String, targets : String, sources : String)
+PortProperty() : SimpleStringProperty
+setPort(Port : SimpleStringProperty) : void
+RecordsProperty() : SimpleStringProperty
+setRecords(Records : SimpleStringProperty) : void
+TargetsProperty() : SimpleStringProperty
+setTargets(Targets : SimpleStringProperty) : void
+SourcesProperty() : SimpleStringProperty
+setSources(Sources : SimpleStringProperty) : void

**ParsingException**
(org::jimmycollins::networktraffic)
+ParsingException()
+ParsingException(message : String)

**TopDestinationIpAddresses**
(org::jimmycollins::networktraffic::util)
-DestinationIpAddresses : ArrayList
+Initialize() : void
+GetData(Flows : ArrayList<Flow>) : Map<String, Integer>

**DatabaseUtil**
(org::jimmycollins::networktraffic::util)
+Query(query : String) : ResultSet
+PersistAnalysisRecord(query : String) : int
+PersistPortData(analysisId : int, table : String, data : Map<String, Integer>) : void
+PersistIpData(analysisId : int, table : String, data : Map<String, Integer>) : void

**TopDestinationPorts**
(org::jimmycollins::networktraffic::util)
-DestinationPorts : ArrayList
+Initialize() : void
+GetData(Flows : ArrayList<Flow>) : Map<String, Integer>

**TopData**
(org::jimmycollins::networktraffic::model)
+Initialize() : void
+GetData(Flows : ArrayList<Flow>) : Map<String, Integer>
+GetTopData(Flows : ArrayList<Flow>) : Map<String, Integer>

**TopSourceIpAddresses**
(org::jimmycollins::networktraffic::util)
-SourceIpAddresses : ArrayList
+Initialize() : void
+GetData(Flows : ArrayList<Flow>) : Map<String, Integer>

**TopSourcePorts**
(org::jimmycollins::networktraffic::util)
-SourcePorts : ArrayList
+Initialize() : void
+GetData(Flows : ArrayList<Flow>) : Map<String, Integer>

**Flow**
(org::jimmycollins::networktraffic::model)
-Date : LocalDateTime
-SourceHost : InetAddress
-DestinationHost : InetAddress
-SourcePort : int
-DestinationPort : int
-FlowType : Enum
+Flow(type : Enum)
+SetDate(date : LocalDateTime) : void
+SetSourceHost(host : InetAddress) : void
+SetDestinationHost(host : InetAddress) : void
+SetSourcePort(port : int) : void
+SetDestinationPort(port : int) : void
+GetSourceHost() : InetAddress
+GetDestinationHost() : InetAddress
+GetSourcePort() : int
+GetDestinationPort() : int
+GetFlowType() : String
+GetDate() : LocalDateTime