

### rootJS Architecture

```

classDiagram
    class Application {
        <<Singleton>>
        -initialized: bool
        -instance: NodeApplication
        -rootJS: Persistent<Object>
        +ctorCallback(args: FunctionCallbackInfo<Value>)
        +staticCtorCallback(args: FunctionCallbackInfo<Value>)
        +memberGetterCallback(property: Local<String>, info: PropertyCallbackInfo<Value>)
        +memberSetterCallback(property: Local<String>, value: Local<Value>, info: PropertyCallbackInfo<Value>)
        +memberFunctionCallback(args: FunctionCallbackInfo<Value>)
        +staticGetterCallback(property: Local<String>, info: PropertyCallbackInfo<Value>)
        +staticSetterCallback(property: Local<String>, value: Local<Value>, info: PropertyCallbackInfo<Value>)
        +staticFunctionCallback(args: FunctionCallbackInfo<Value>)
        +Initialize(exports: Local<Object>, module: Local<Object>)
        -Exit(args: void*)
        +Instance(): NodeApplication
        -NodeApplication(acn: char*, argc: int*, argv: char**) «constructor»
        -initROOTGlobals()
        -initROOTMessageCallback()
        -exposeROOT()
        -exposeMacros()
        -exposeClasses()
        -exposeClass(clazz: TClassRef)
        +getIsolate(): Isolate*
        +getExports(): Local<Object>
        +getTemplateFactory(): TemplateFactory
        +getFunctionFactory(): ProxyFunctionFactory
        +getObjectFactory(): ProxyObjectFactory
    }
    class ROOT {
        <<Singleton>>
    }
    class ClassHelper {
        +IsNamespace(scope: TCppScope): bool
        +IsAbstract(klass: TCppType): bool
        +IsEnum(type_name: string): bool
        +IsStruct(type_name: string): bool
        +GetFinalName(klass: TCppType): string
        +GetScopedFinalName(klass: TCppType): string
        +GetNumBases(klass: TCppType): TCplIndex
        +GetBaseName(klass: TCppType, ibase: TCplIndex): string
        +IsSubtype(derived: TCppType, base: TCppType): bool
        +GetNumDatamembers(scope: TCppScope): TCplIndex
        +GetDatamemberName(scope: TCppScope, idata: TCplIndex): string
        +GetDatamemberType(scope: TCppScope, idata: TCplIndex): string
        +GetDatamemberOffset(scope: TCppScope, idata: TCplIndex): ptrdiff
        +GetDatamemberIndex(scope: TCppScope, name: string): TCplIndex
        +IsPublicData(scope: TCppScope, idata: TCplIndex): bool
        +IsStaticData(scope: TCppScope, idata: TCplIndex): bool
        +IsConstData(scope: TCppScope, idata: TCplIndex): bool
        +IsEnumData(scope: TCppScope, idata: TCplIndex): bool
        +resolveAddress(staticMember: TDataMember, clazz: TClassRef): void*
        +resolveAddress(staticMember: TDataMember, clazz: TClassRef): void*
    }
    class NodeApplication {
        <<Singleton>>
        -initialized: bool
        -instance: NodeApplication
        -rootJS: Persistent<Object>
        +ctorCallback(args: FunctionCallbackInfo<Value>)
        +staticCtorCallback(args: FunctionCallbackInfo<Value>)
        +memberGetterCallback(property: Local<String>, info: PropertyCallbackInfo<Value>)
        +memberSetterCallback(property: Local<String>, value: Local<Value>, info: PropertyCallbackInfo<Value>)
        +memberFunctionCallback(args: FunctionCallbackInfo<Value>)
        +staticGetterCallback(property: Local<String>, info: PropertyCallbackInfo<Value>)
        +staticSetterCallback(property: Local<String>, value: Local<Value>, info: PropertyCallbackInfo<Value>)
        +staticFunctionCallback(args: FunctionCallbackInfo<Value>)
        +Initialize(exports: Local<Object>, module: Local<Object>)
        -Exit(args: void*)
        +Instance(): NodeApplication
        -NodeApplication(acn: char*, argc: int*, argv: char**) «constructor»
        -initROOTGlobals()
        -initROOTMessageCallback()
        -exposeROOT()
        -exposeMacros()
        -exposeClasses()
        -exposeClass(clazz: TClassRef)
        +getIsolate(): Isolate*
        +getExports(): Local<Object>
        +getTemplateFactory(): TemplateFactory
        +getFunctionFactory(): ProxyFunctionFactory
        +getObjectFactory(): ProxyObjectFactory
    }
    class ROOT {
        <<Singleton>>
    }
    class ClassHelper {
        +IsNamespace(scope: TCppScope): bool
        +IsAbstract(klass: TCppType): bool
        +IsEnum(type_name: string): bool
        +IsStruct(type_name: string): bool
        +GetFinalName(klass: TCppType): string
        +GetScopedFinalName(klass: TCppType): string
        +GetNumBases(klass: TCppType): TCplIndex
        +GetBaseName(klass: TCppType, ibase: TCplIndex): string
        +IsSubtype(derived: TCppType, base: TCppType): bool
        +GetNumDatamembers(scope: TCppScope): TCplIndex
        +GetDatamemberName(scope: TCppScope, idata: TCplIndex): string
        +GetDatamemberType(scope: TCppScope, idata: TCplIndex): string
        +GetDatamemberOffset(scope: TCppScope, idata: TCplIndex): ptrdiff
        +GetDatamemberIndex(scope: TCppScope, name: string): TCplIndex
        +IsPublicData(scope: TCppScope, idata: TCplIndex): bool
        +IsStaticData(scope: TCppScope, idata: TCplIndex): bool
        +IsConstData(scope: TCppScope, idata: TCplIndex): bool
        +IsEnumData(scope: TCppScope, idata: TCplIndex): bool
        +resolveAddress(staticMember: TDataMember, clazz: TClassRef): void*
        +resolveAddress(staticMember: TDataMember, clazz: TClassRef): void*
    }
    class ProxyObjectFactory {
        +createProxyObject(type: TDataMember, holder: ProxyObject): ProxyObject
    }
    class TemplateFactory {
        -cache: map<string, Persistent<FunctionTemplate>>
        +createTemplate(clazz: TClassRef): Local<FunctionTemplate>
    }
    class ProxyFunctionFactory {
        +createProxyFunction(info: TMethod): ProxyFunction
        +fromArgs(name: string, clazz: TClassRef, args: FunctionCallbackInfo): ProxyFunction
    }
    class ProxyObject {
        #address: void*
        #type: TDataMember
        #ProxyObject(address: void*, type: TDataMember) «constructor»
        +getAddress(): void*
        +getType(): TDataMember
        +set(value: ProxyObject)
        +get(): Local<Value>
        +isPrimitive(): bool
    }
    class ProxyFunction {
        -address: void*
        -info: TFunction
        +ProxyFunction(address: void*, info: TFunction) «constructor»
        +convertArgs(args: FunctionCallbackInfo): ProxyObject[]
        +call(args: ProxyObject[]): ProxyObject
        +isTemplateFunction(): bool
    }
    class TemplateProxy
    class EnumProxy
    class StructProxy
    class ArrayProxy
    class PointerProxy
    class ReferenceProxy
    class PrimitiveProxy
    class NumberProxy
    class StringProxy
    class BooleanProxy
    class nodeObjectWrap {
        <<Singleton>>
    }

    Application <|-- NodeApplication
    NodeApplication <|-- ROOT
    NodeApplication --> ClassHelper
    NodeApplication --> ProxyObjectFactory : 1
    NodeApplication --> TemplateFactory : 1
    NodeApplication --> ProxyFunctionFactory : 1
    ProxyObjectFactory ..> ProxyObject : creates
    ProxyFunctionFactory ..> ProxyFunction : creates
    ProxyObject <|-- TemplateProxy
    ProxyObject <|-- EnumProxy
    ProxyObject <|-- StructProxy
    ProxyObject <|-- ArrayProxy
    ProxyObject <|-- PointerProxy
    ProxyObject <|-- ReferenceProxy
    ProxyObject <|-- PrimitiveProxy
    PrimitiveProxy <|-- NumberProxy
    PrimitiveProxy <|-- StringProxy
    PrimitiveProxy <|-- BooleanProxy
    ProxyObject --> nodeObjectWrap
    
```

The diagram illustrates the rootJS Architecture, showing the relationships between various classes and their components.

**Application** (Singleton) is the root class, which inherits from **NodeApplication** (Singleton). It contains a **ROOT** (Singleton) object. The **Application** class has a **ctorCallback** method and a **staticCtorCallback** method. It also has a **memberGetterCallback** method and a **memberSetterCallback** method. The **Application** class has a **staticFunctionCallback** method and a **staticGetterCallback** method. It also has a **staticSetterCallback** method and a **staticFunctionCallback** method. The **Application** class has a **Initialize** method and a **Exit** method. The **Application** class has a **Instance** method and a **NodeApplication** constructor. The **Application** class has a **initROOTGlobals** method and a **initROOTMessageCallback** method. The **Application** class has a **exposeROOT** method and a **exposeMacros** method. The **Application** class has a **exposeClasses** method and a **exposeClass** method. The **Application** class has a **getIsolate** method and a **getExports** method. The **Application** class has a **getTemplateFactory** method and a **getFunctionFactory** method. The **Application** class has a **getObjectFactory** method.

**ClassHelper** is a class that provides utility functions for working with C++ types. It includes methods like **IsNamespace**, **IsAbstract**, **IsEnum**, **IsStruct**, **GetFinalName**, **GetScopedFinalName**, **GetNumBases**, **GetBaseName**, **IsSubtype**, **GetNumDatamembers**, **GetDatamemberName**, **GetDatamemberType**, **GetDatamemberOffset**, **GetDatamemberIndex**, **IsPublicData**, **IsStaticData**, **IsConstData**, **IsEnumData**, **resolveAddress**, and **resolveAddress**.

**ProxyObjectFactory** is a class that provides a **createProxyObject** method, which takes a **TDataMember** and a **ProxyObject** as input and returns a **ProxyObject**.

**TemplateFactory** is a class that provides a **createTemplate** method, which takes a **TClassRef** as input and returns a **Local<FunctionTemplate>**. It also has a **cache** attribute of type **map<string, Persistent<FunctionTemplate>>**.

**ProxyFunctionFactory** is a class that provides a **createProxyFunction** method, which takes a **TMethod** as input and returns a **ProxyFunction**. It also has a **fromArgs** method, which takes a **string**, a **TClassRef**, and a **FunctionCallbackInfo** as input and returns a **ProxyFunction**.

**ProxyObject** is a class that represents a proxy object. It has attributes **address** (void\*) and **type** (TDataMember). It has a constructor **ProxyObject(address: void\*, type: TDataMember)** and methods **getAddress**, **getType**, **set**, **get**, and **isPrimitive**.

**ProxyFunction** is a class that represents a proxy function. It has attributes **address** (void\*) and **info** (TFunction). It has a constructor **ProxyFunction(address: void\*, info: TFunction)** and methods **convertArgs**, **call**, and **isTemplateFunction**.

The diagram also shows several proxy classes: **TemplateProxy**, **EnumProxy**, **StructProxy**, **ArrayProxy**, **PointerProxy**, **ReferenceProxy**, **PrimitiveProxy**, **NumberProxy**, **StringProxy**, and **BooleanProxy**. These classes inherit from **ProxyObject** or **ProxyFunction**.

**node::ObjectWrap** is a class that provides a **Wrap** method, which takes a **ProxyObject** as input and returns a **node::ObjectWrap**.

