



DelphiDay

italian conference

LOGIFY

Meta-Logger for Delphi



PAOLO ROSSI

WINTech ITALIA - CTO

SENCHA & EMB. MVP



 blog.paolorossi.net

 paolo@paolorossi.net

 twitter.com/awebguy

 github.com/paolo-rossi

 linkedin.com/in/paolo-rossi-pc



DelphiDay
italian conference

19-20 Giugno 2025
Piacenza



wintech
italia



GITHUB PROJECTS



github.com/paolo-rossi



Delphi JWT

JSON Web Token Library



WiRL

REST Library for Delphi



Linux Daemon

Real Linux daemons



Delphi Neon

JSON Serialization Library



OpenAPI-Delphi

OpenAPI 3.0 Library



NATS Delphi

NATS Client Library for Delphi



AGENDA

1. Logging: why, when, what, where, how
2. Challenges of the standard approach
3. Introducing Logify
4. Compare your code w/ and w/o Logify
5. Logify architecture & internals



Logging

1



WHY



Debugging and Troubleshooting

- Detailed record of an application's execution
- Easier to trace the root cause of errors and unexpected behavior



Performance Monitoring

- Capture performance-related data
- Identify bottlenecks and optimize application performance



Security Auditing

- Record user actions, system events, and potential security threats
- identify and mitigate risks



WHY

- ★ Communication and Collaboration
 - Single source of truth
 - Efficient communication between developers and users
- ★ Troubleshooting Intermittent Issues
 - Capture details of intermittent errors difficult to reproduce
- ★ Understanding Application Behavior
 - Insights into how an application is used
- ★ Facilitating Fast Incident Response
 - Quickly identify the cause of incidents and take appropriate action



WHEN

- ★ Exceptions
 - try-except, OnException
- ★ (Important) parameters check
- ★ External (API, DLL) function return



WHEN NOT



Random, useless log calls

- Like comments



WHERE (SHORT VERSION)



Local File



WHERE (SORTED BY LOG RELIABILITY)

1. Local File
2. EventLog
3. Database
4. Network File
5. Network Server (TCP, HTTP, etc...)
6. Cloud (S3, Blob Storage, etc...)
7. Message Service (Slack, Telegram, etc...)



WHAT

- ★ Write meaningful log entries
 - Context is important
- ★ Use log levels correctly
- ★ Make a distinction between signal and noise
 - Logging too much is bad as no logging



HOW



KISS principle

- `Log(string)`
- `Log(string, level)`
- `Log(Exception, string)`
- `LogInfo(string)`, `LogDebug(string)`, etc...



That's it! that's the (only) job for a Logger



No “high level” overloads

- `Objects`, `DataSets`, etc...



LOGGING: WHAT IS NOT

- ★ Writing detailed performance info
 - Measuring/Benchmarking
- ★ Writing detailed metrics (CPU%, Memory, I/O, etc)
 - Monitoring (Prometheus, etc...)
- ★ Writing detailed memory usage
 - Memory management/info (FastMM, madExcept, etc...)
- ★ Writing function flow
 - Tracing/Benchmarking
- ★ Writing detailed user operations (on database)
 - Auditing



The “Delphi” way 2



DELPHI & LOGGING

- ★ Delphi does not have a standard logging library
 - Codesite?
 - Several Logging Libraries commercial and open source
- ★ Delphi does not have an official Logging API (interface)
- ★ Logify tries to rectify this!



THE CLASSIC APPROACH

- ★ Pick a logging library
- ★ Learn the library own “API”
- ★ Start to write log commands according to the library



PROS

- ★ Pretty straightforward approach
- ★ Usually, libraries expose a rich set of functions to interact with the log
- ★



CONS

- ★ What if you want to change the library later on?
- ★ Well you must remove all the code (for the log part)
- ★ And restart the process
 - Pick a new logging library
 - Learn the library own “API”
 - Start to write log commands according to the library



Logify

3



Logify Repository



LOGIFY

- ★ Main Goal: to provide a neutral API for logging
- ★ Open source project (MIT license)
- ★ Takes after the .NET logging interface
 - Same concepts as Java as well



LOGIFY

- ★ Logify is not a logging library
 - Although there are some simple loggers
 - Mostly written as an example
 - The file logger is quite good, though
- ★ Logify does not write the log itself
 - Your (usual) logging library does
- ★ You log using the Logify interface
- ★ You can “attach” to Logify your Logging Library
 - Key word can...



FEATURES

- ★ No dependency other than Logify.pas
- ★ Dummy logger implementation
 - You can use it without any real logger
- ★ No Direct Logger Class Dependency
- ★ True Decoupling
- ★ Future-Proofing



FEATURES

**We can say that the main feature of
Logify is to do... nothing 😁**



Comparing Code

4



Classic approach

- ★ Classic approach sample log
- ★ Using it in another application
- ★ Changing the logging library



Logify approach

- ★ Logify sample log
- ★ Using it in another application
- ★ Changing the logging library



Logify Internals

5



ILogger interface

- ★ Importing Logify.pas
- ★ Using Logger utility singleton



ILoggerAdapter interface

- ★ Logify ↔ Your Logger
- ★ Creation & Logger Configuration
- ★ The TLoggerAdapterHelper utility class
 - function FormatMsg()
 - procedure InternalLog()
 - procedure InternalRaw()

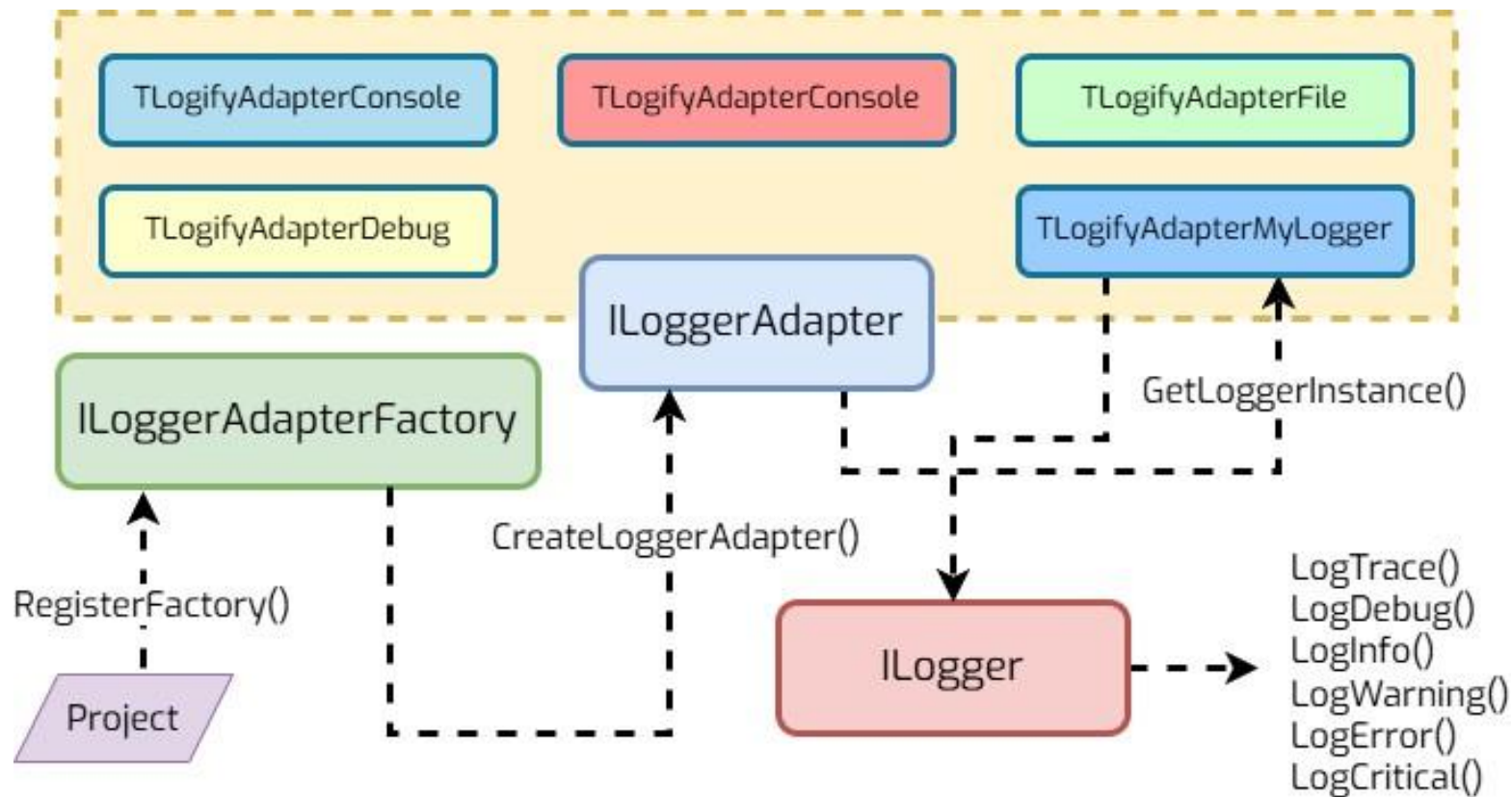


ILoggerAdapterFactory interface

- ★ Creation of the adapter
- ★ Factory for DebugLogger & FileLogger



LOGIFY ARCHITECTURE





THANK YOU