LOCALIZATION ENGINEERING • CLASS 1 • MODULE 3

# How Software Organizes Human Ideas

Mastering key-value pair mechanics, protecting dynamic variable placeholders, and utilizing targeted regular expressions for syntax audits.

SYNTAX: STRUCTURAL DATA & REGEX

LESSON 3.1 • CORE CONCEPTS

# Separating Labels from Meaning

Localization files decouple technical identifiers from human-visible strings to ensure seamless software delivery.

**The Technical Key**
Immutable identifier referenced by software engine. Never translate.

**The Localized Value**
Human-facing natural language displayed on screen. Safe to modify.

```
// KEY-VALUE HINGE EXAMPLE
{
  "btn_confirm": "Confirm Purchase",
  "lbl_welcome": "Welcome back!"
}
```

LESSON 3.1 • VARIABLE SAFETY

# Protecting Dynamic Runtime Variables

### Positional Numbers

{0} • {1}

Preserve numeric order across target language grammar changes.

### Named Identifiers

{username}

Never translate inner variable terms inside bracket boundaries.

### Printf Specifiers

%s • %d

Maintain exact type specifiers to prevent engine crashes.

LESSON 3.2 • STRUCTURAL RULES

# Structural Punctuation & Escape Sequences

## Quote Boundaries

Outer double quotes define string limits. Internal spoken quotes must be escaped with a backslash (\") to avoid syntax breaks.

## Comma & Colon Mechanics

Colons separate keys from values. Trailing commas after final key-value pairs are invalid in JSON standard and break parsers.

LESSON 3.2 • TARGETED SEARCH

# Human-Centered Regex Patterns

`(?`

Spot Unescaped Quotes

`\{[0-9]+\}`

Verify Numeric Placeholders

`[A-Za-z0-9]+`

Flag Untranslated ASCII

HANDS-ON LAB • STEP-BY-STEP

# Lab 3: Meaning & Logic Alignment

1. **Audit Syntax:** Run (? regex pattern in VS Code to locate unescaped quotes.
2. **Repair Escape Slashes:** Add backslashes to internal quotes without breaking string limits.
3. **Restore Variables:** Correct translated variable term {poziom} back to engine parameter {username}.

MODULE 3 COMPLETE

# Ready for Reader Comfort Math?

With structural code and placeholders secure, Module 4 moves into timecodes, CPS reading speeds, and language expansion math.

NEXT: MODULE 4 — RESPECTING THE READER'S TIME & EYES