1. Involved Classes

Class Hierarchy Role

CulturalContext Context ➔ CulturalContext Represents cultural design constraints (e.g., language directionality).

TextInputComponent UIComponent ➔ InputComponent ➔ TextInputComponent Models UI elements for text input (e.g., single-line, multi-line).

2. Object Properties

Property Domain Range Purpose

suitableFor TextInputComponent CulturalContext Links a text input component to a cultural context it supports.

3. Data Properties

Property Domain Range Purpose

hasLanguageInputDirection CulturalContext xsd:string Specifies the language direction ("LTR" or "RTL") for a cultural context.

4. Instances

Instance Class Data Property Assertion

ArabicContext CulturalContext hasLanguageInputDirection "RTL"

SingleLineInput TextInputComponent suitableFor ArabicContext

5. Connections

Visual Representation

[TextInputComponent] ── suitableFor ──→ [CulturalContext]

│

└─ hasLanguageInputDirection "RTL"

Explanation:

The SingleLineInput (a TextInputComponent) is linked to ArabicContext (a CulturalContext) via suitableFor.

ArabicContext is tagged with hasLanguageInputDirection "RTL" to denote right-to-left language support.

6. Validation Steps

SPARQL Query:

sparql

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

PREFIX owl: <http://www.w3.org/2002/07/owl#>

PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX onto: <http://www.semanticweb.org/asifcomputer/ontology/2025/ui\_ux\_automation#>

SELECT ?comp

WHERE {

?comp a onto:TextInputComponent ;

onto:suitableContextFor ?ctx .

?ctx a onto:CulturalContext ;

onto:hasLanguageInputDirection "RTL" .

}Result: SingleLineInput.

Reasoner Check:

InputComponent and

(suitableFor some (

CulturalContext and

(hasLanguageInputDirection value "RTL")

))

* **Meaning**:
  + A TextInputComponent is **exactly** an InputComponent that is suitableFor a CulturalContext with hasLanguageInputDirection = "RTL".

**Inferred Classification**:

* The reasoner will classify SingleLineInput as a TextInputComponent because it is linked to ArabicContext (which has hasLanguageInputDirection "RTL").

**Reasoning & Automation Summary**

**What Reasoning Achieves**:  
By defining TextInputComponent with **necessary and sufficient conditions** linked to RTL cultural contexts, the ontology automatically infers:

* Any UI component linked (via suitableFor) to a CulturalContext with hasLanguageInputDirection = "RTRL" is **RTL-compliant**.
* This eliminates manual tagging, as the reasoner classifies components dynamically.

**How This Solves Client Requirements**:

1. **Cultural Adaptation**:
   * Ensures UI components (e.g., text inputs) automatically adapt to cultural constraints (e.g., Arabic RTL layouts).
   * Clients can deploy apps globally without manual reconfiguration.
2. **Consistency**:
   * Guarantees all RTL-compliant components are identified programmatically, avoiding human error.
3. **Efficiency**:
   * Reduces time spent manually auditing UI components for cultural compatibility.

**Automation in UI/UX Design**:

* **Dynamic Component Selection**:  
  Tools can query the ontology to auto-select RTL-compliant components for Arabic/UIs, streamlining design workflows.
* **Design System Governance**:  
  Ensures compliance with accessibility and localization standards (e.g., WCAG, regional guidelines).

**In One Line**:

**This ontology automates culturally-aware UI/UX design by dynamically linking components to language/cultural constraints, ensuring global usability with minimal manual effort.** 🌍✨