GenesisDecoder — HTML view of Java source

A browsable, copyable HTML representation of your `GenesisDecoder` Java class.

File: **GenesisDecoder.java** Language: Java Converted: HTML view

Below is the full source. Use the Copy button to copy the code to clipboard.

Copy code Download .java

```java
public class GenesisDecoder {

// --- BRAIN LANGUAGE DATABASE ---
// This is a massive database built from the mappings on Pages 58-93.
// Example: The "Thought" object for "Clap" would be linked to the sequence:
// 1. Signal from Right-Occipital sensor cluster (Right Back Side)
// 2. Vibration in Right-Tympanic sensor (Right Ear)
// 3. Activation in Right/Left-Ulna sensors (Elbows)
// 4. Acoustic signature of hands clapping + seismic "downward escape" vibration in feet sensors.

// 5. Inertial measurement of "shoulder deflation".
private Map brainLanguageMap = new HashMap<>();
// Populate this map from the book's definitions...
// brainLanguageMap.put("CLAP_SEQUENCE", new Thought("Clap", "👏", "Action"));

// --- DECODING PIPELINE ---
```

```
public Thought decodeFullBodySignal(BodyDataStream bodyData) {

    // STEP 1: MIRRORING & SYNCHRONIZATION (Page 40-42)
    // The system calibrates by establishing a baseline "mirror state" for the user.
    // It looks for the unique signature of the "Right Back Side" initial activation.
    if (!isLimbicSignalInitiated(bodyData)) {
        return new Thought("Idle", "☁️", "Baseline");
    }

    // STEP 2: PATTERN RECOGNITION & SEQUENCE EXTRACTION
    // Analyzes the sensor data to reconstruct the precise sequence of micro-movements
    // and vibrations as defined by the Brain Language.
    String extractedSequence = extractSequenceFromSensors(bodyData);

    // STEP 3: PREDICTIVE TEXT ALGORITHM (Page 27-28, 37-38)
    // Uses the assigned alphabetical/numeric positions on the body to predict
    // the likely word or sentence being formed before the sequence is complete.
    String predictedThought = predictiveAlgorithm.predict(extractedSequence);

    // STEP 4: CORE DECODING (The provided code, enhanced)
    // Matches the extracted and predicted sequence to the Brain Language database.
    Thought decodedThought = brainLanguageMap.get(extractedSequence);

    // If no direct match, use the original impulse pattern as a fallback (less reliable)
    if (decodedThought.archetype.equals("Unmapped")) {
        decodedThought = decodeImpulsePattern(bodyData.getRawEMSignal());
    }

    return decodedThought;
}

// --- The Original Impulse Decoder (Now a fallback method) ---
private Thought decodeImpulsePattern(String rawImpulse) {
    if (rawImpulse.contains("Δα7")) return new Thought("Curiosity", "🌀", "Explorer");
    if (rawImpulse.contains("βγ3")) return new Thought("Fear", "⚠️", "Guardian");
    if (rawImpulse.contains("Ωλ2")) return new Thought("Joy", "🌈", "Creator");
```

```
    if (rawImpulse.contains("Ψπ9")) return new Thought("Legacy", "🔗", "Architect");
    return new Thought("Unknown", "❓", "Unmapped");
}

public static class Thought {
    public final String symbol; // The word: "Clap", "Beauty"
    public final String glyph; // The emoji/symbol: 👏, 😍
    public final String archetype; // The category: "Action", "Emotion"

    public Thought(String symbol, String glyph, String archetype) {
        this.symbol = symbol;
        this.glyph = glyph;
        this.archetype = archetype;
    }
}

// ... Helper methods for sensor data analysis, sequence extraction, and prediction.

}
```

Tip: Click "Download .java" to save the file locally, or "Copy code" to paste it into your IDE.