HSML Conversion & Autonomous Waypoint Navigation

Presented by:
Diego Cordova

Internship Program:
Jet Propulsion Laboratory (JPL) — SIRI Program

Mentors:
Dr. Thomas Lu
Dr. Edward (Ed) Chow

Introduction / Overview

Hello, I am Diego Cordova.

-**HSML Conversion Tool:** Automating conversion of arbitrary JSON data into structured HSML JSON using AI, enabling standardized data sharing.

-Autonomous Waypoint Navigation System: Developing a modular, reusable system in Unity for simulated lunar rovers to autonomously follow waypoint paths with realistic motion.

- These projects aim to enhance interoperability and realism across multi-platform simulations, supporting NASA’s mission for advanced rover autonomy and data integration.

HSML Conversion

- **Purpose:** Convert arbitrary JSON inputs into structured HSML JSON objects using the OpenAI API.
- **Main steps:**
  - Initialize the OpenAI client with credentials.
  - Create and index a vector store for the HSML schema document.
  - Update the assistant to leverage the schema via the file_search tool.
  - Send JSON payloads to the assistant and receive HSML-formatted output.

Why HSML?

- Provides a unified schema for diverse data types (Agents, Activities, Credentials).
- Simplifies downstream processing, validation, and integration with other HSML-enabled tools.

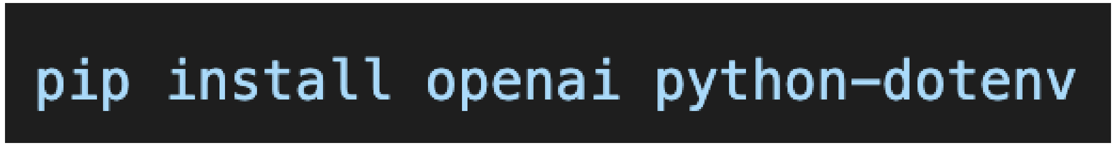
Project Context & Benefits

- **Use Case:** Automate ingestion of user-generated JSON events into an HSML-based knowledge graph.
- **Benefits:**
  - **Scalability:** Handle arbitrary JSON shapes without manual mapping.
  - **Maintainability:** Centralized schema updates (just update HSML document).
  - **Consistency:** Enforces field presence (@context, swid, type, etc.) across all data.

Why Integration Matters

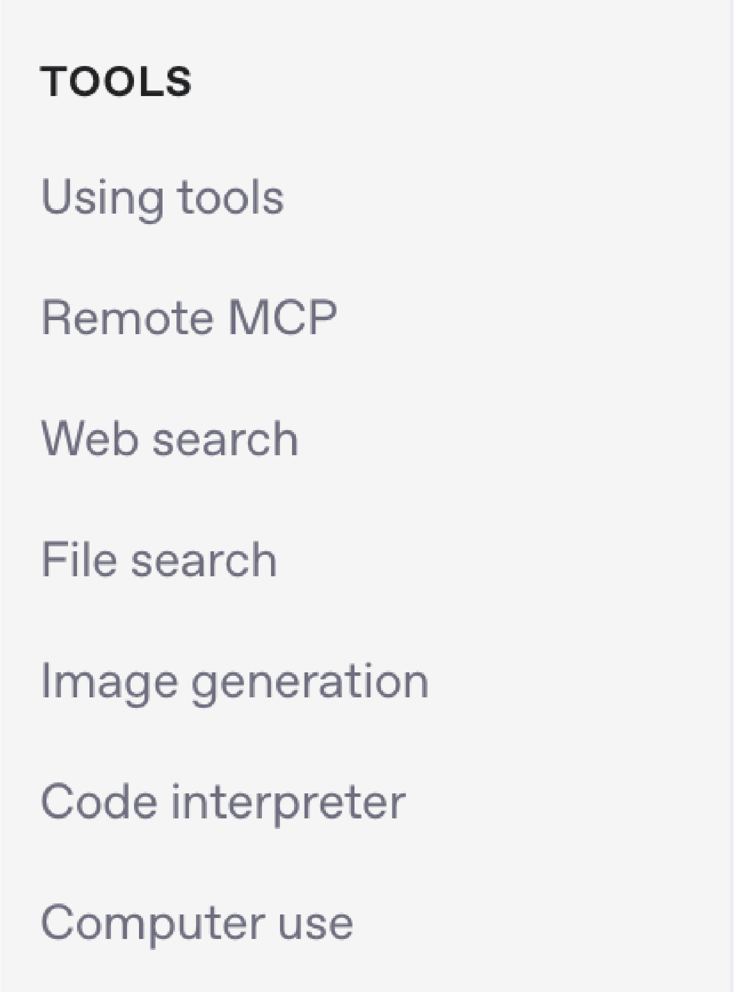
- **Cross-Industry Collaboration:** HSML acts as a lingua franca, allowing organizations with different internal data formats to exchange models, agents, and services seamlessly.
- **Shared Ecosystem:** By standardizing on HSML, any HSML-enabled tool (model repositories, agent runtimes, analytics platforms) can interoperate without bespoke adapters.
- **Conversion as Bridge:** Our script is the connector: it translates arbitrary JSON into HSML, enabling data from any source to join the shared ecosystem.

Install Dependencies



```
pip install openai python-dotenv
```

- **openai library**
  - Provides a Python wrapper around the OpenAI REST API.
  - Simplifies authentication, request formatting, and response handling when interacting with models, vector stores, and other OpenAI services.
- **python-dotenv package**
  - Loads environment variables from a local .env file into os.environ at runtime.
  - Keeps secrets (API keys, assistant IDs) out of your source code, improving security and configurability.



TOOLS

Using tools

Remote MCP

Web search

File search

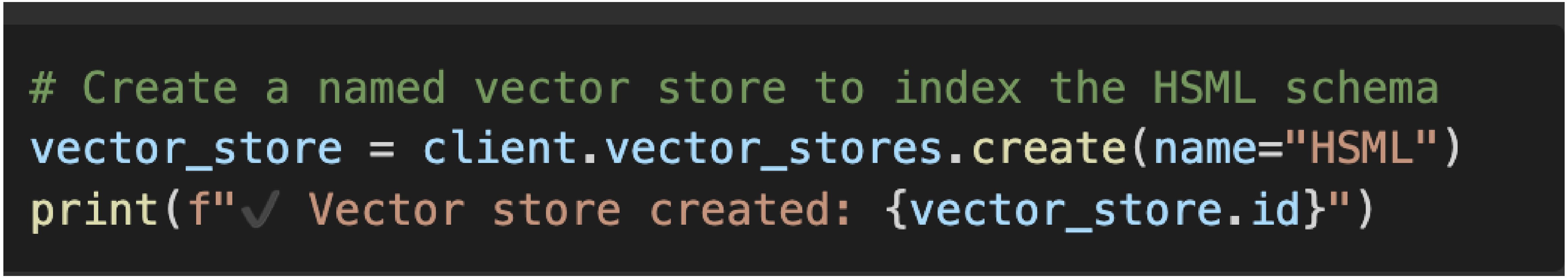
Image generation

Code interpreter

Computer use

Creating the Vector Store

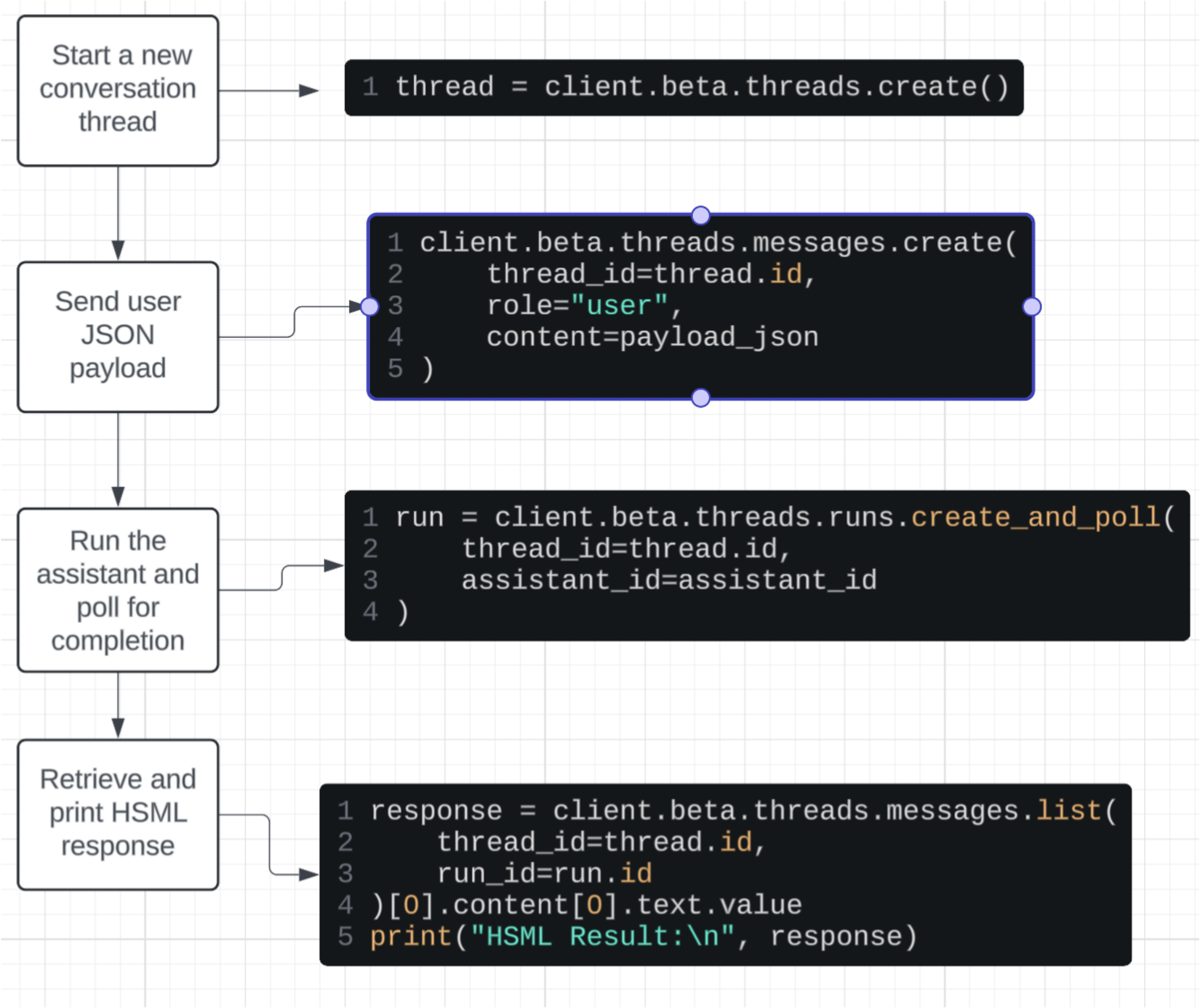
- **Vector store purpose:** Provides semantic search capability so the assistant can dynamically look up field names and rules from the schema.



```
# Create a named vector store to index the HSML schema
vector_store = client.vector_stores.create(name="HSML")
print(f"✓ Vector store created: {vector_store.id}")
```

Conversion Pipeline

- **Threads:** Maintain context for multi-step conversions or follow-up queries.
- **Polling:** Ensures synchronous behavior in a script environment.



Start a new conversation thread

```
thread = client.beta.threads.create()
```

Send user JSON payload

```
client.beta.threads.messages.create(
    thread_id=thread.id,
    role="user",
    content=payload_json
)
```

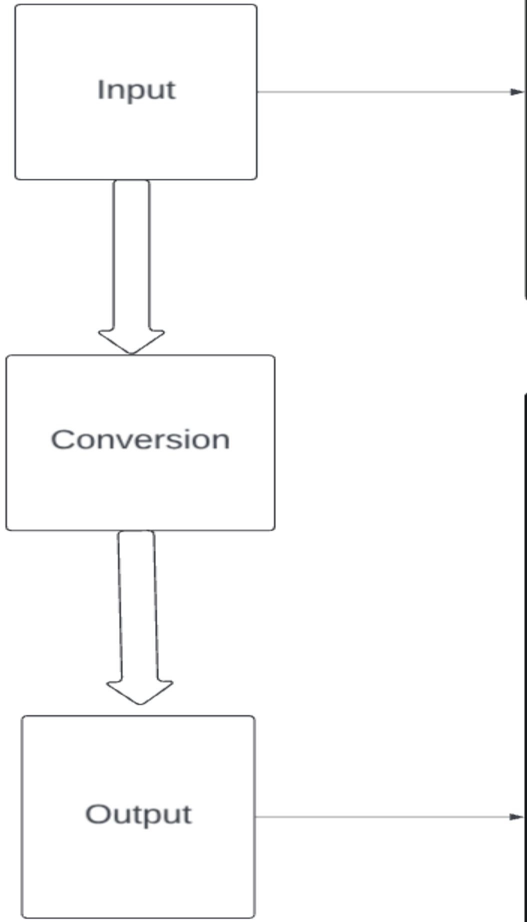
Run the assistant and poll for completion

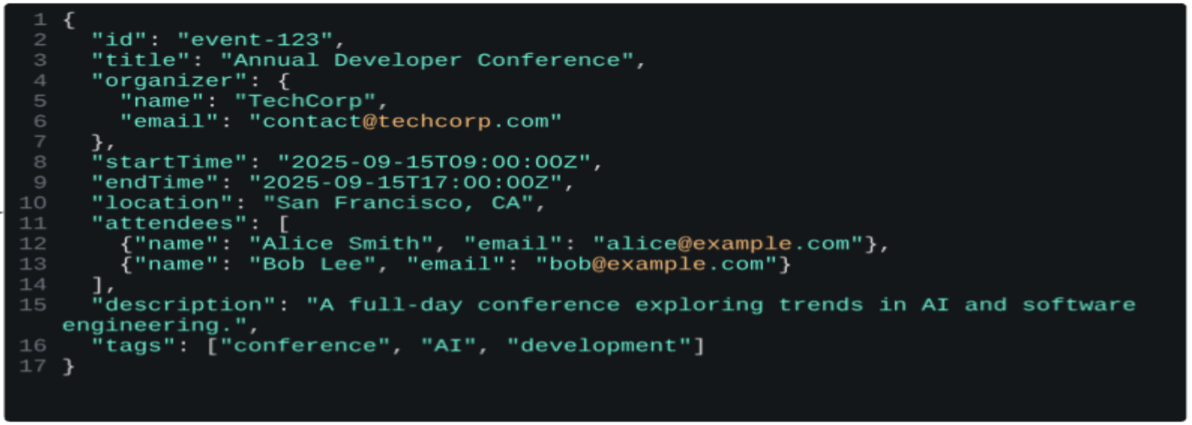
```
run = client.beta.threads.runs.create_and_poll(
    thread_id=thread.id,
    assistant_id=assistant_id
)
```

Retrieve and print HSML response

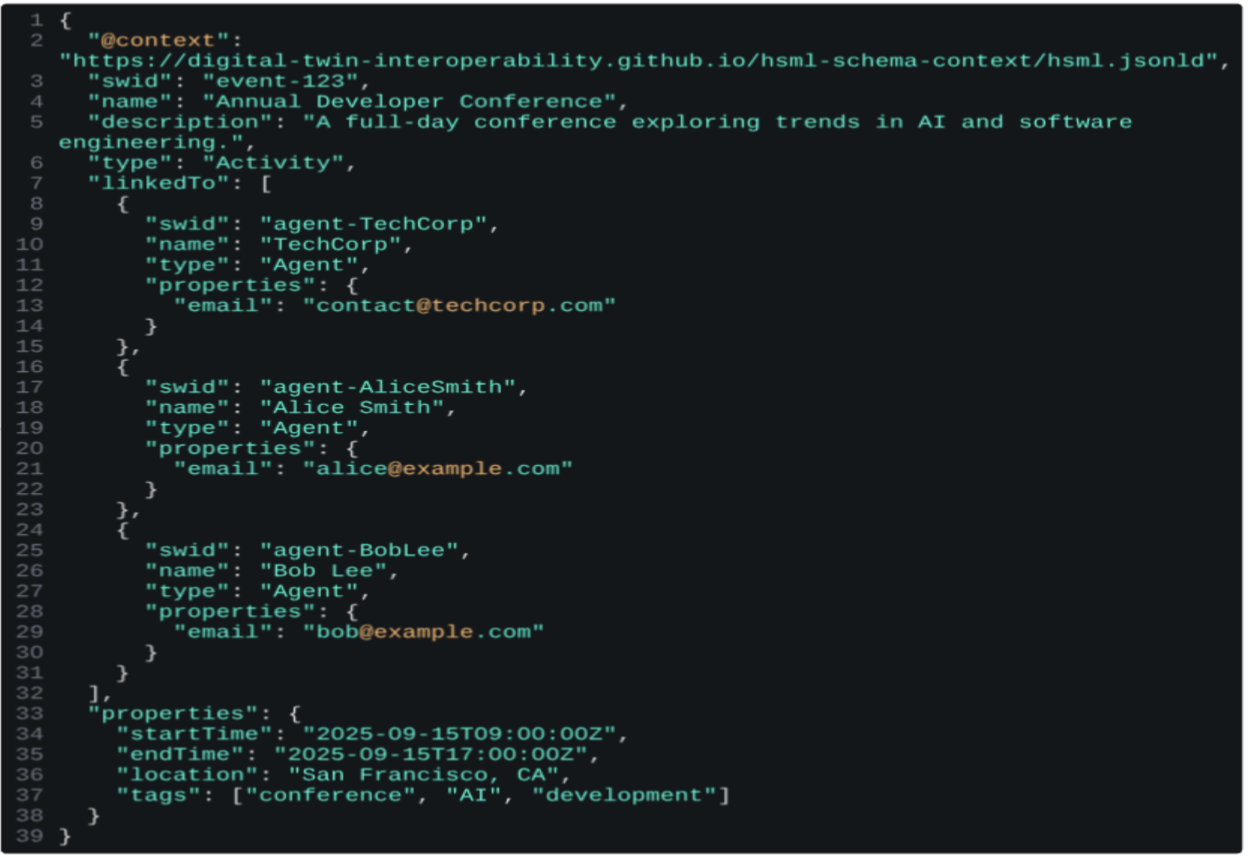
```
response = client.beta.threads.messages.list(
    thread_id=thread.id,
    run_id=run.id
)[0].content[0].text.value
print("HSML Result:\n", response)
```

Example Conversion



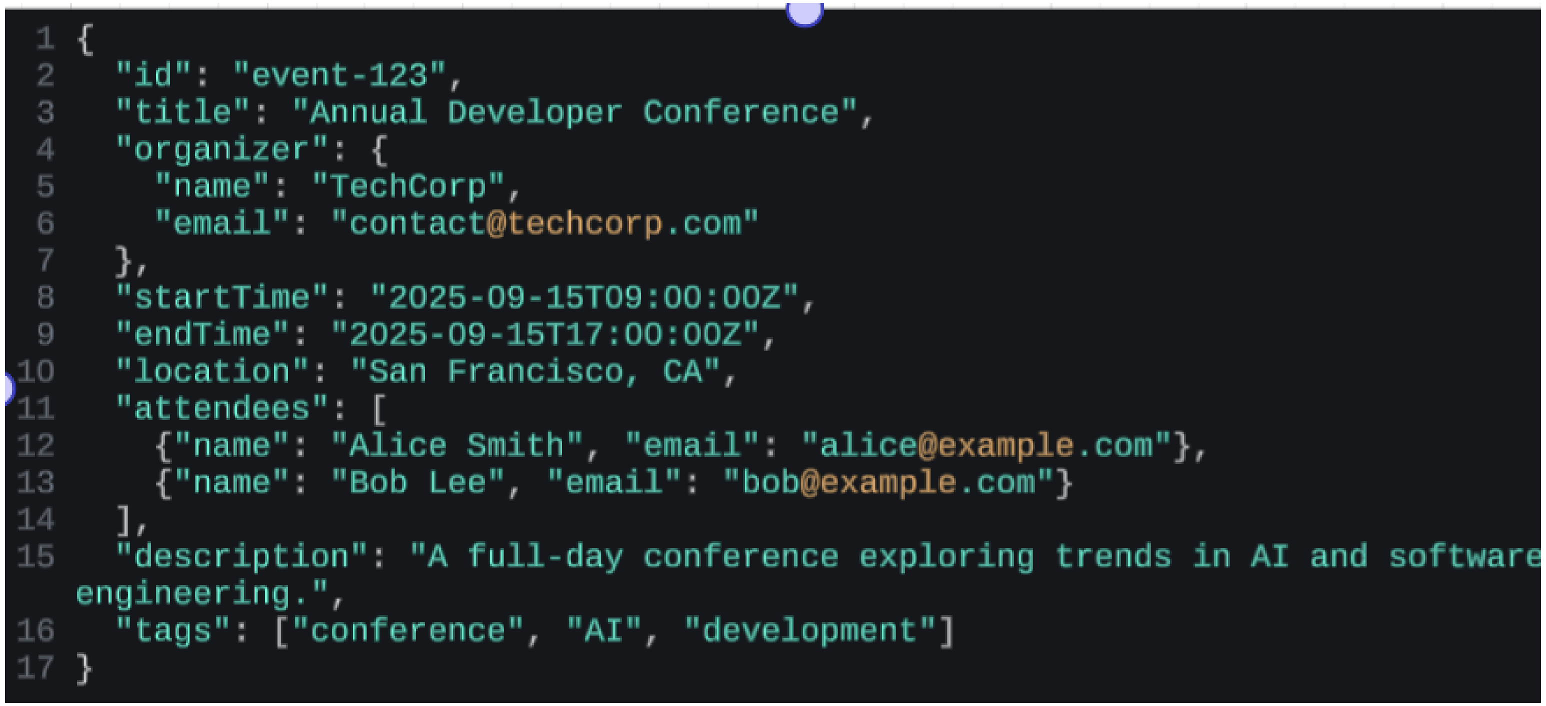


```json
{
  "id": "event-123",
  "title": "Annual Developer Conference",
  "organizer": {
    "name": "TechCorp",
    "email": "contact@techcorp.com"
  },
  "startTime": "2025-09-15T09:00:00Z",
  "endTime": "2025-09-15T17:00:00Z",
  "location": "San Francisco, CA",
  "attendees": [
    {"name": "Alice Smith", "email": "alice@example.com"},
    {"name": "Bob Lee", "email": "bob@example.com"}
  ],
  "description": "A full-day conference exploring trends in AI and software engineering.",
  "tags": ["conference", "AI", "development"]
}
```



```json
{
  "@context": "https://digital-twin-interoperability.github.io/hsml-schema-context/hsml.jsonld",
  "swid": "event-123",
  "name": "Annual Developer Conference",
  "description": "A full-day conference exploring trends in AI and software engineering.",
  "type": "Activity",
  "linkedTo": [
    {
      "swid": "agent-TechCorp",
      "name": "TechCorp",
      "type": "Agent",
      "properties": {
        "email": "contact@techcorp.com"
      }
    },
    {
      "swid": "agent-AliceSmith",
      "name": "Alice Smith",
      "type": "Agent",
      "properties": {
        "email": "alice@example.com"
      }
    },
    {
      "swid": "agent-BobLee",
      "name": "Bob Lee",
      "type": "Agent",
      "properties": {
        "email": "bob@example.com"
      }
    }
  ],
  "properties": {
    "startTime": "2025-09-15T09:00:00Z",
    "endTime": "2025-09-15T17:00:00Z",
    "location": "San Francisco, CA",
    "tags": ["conference", "AI", "development"]
  }
}
```

INPUT

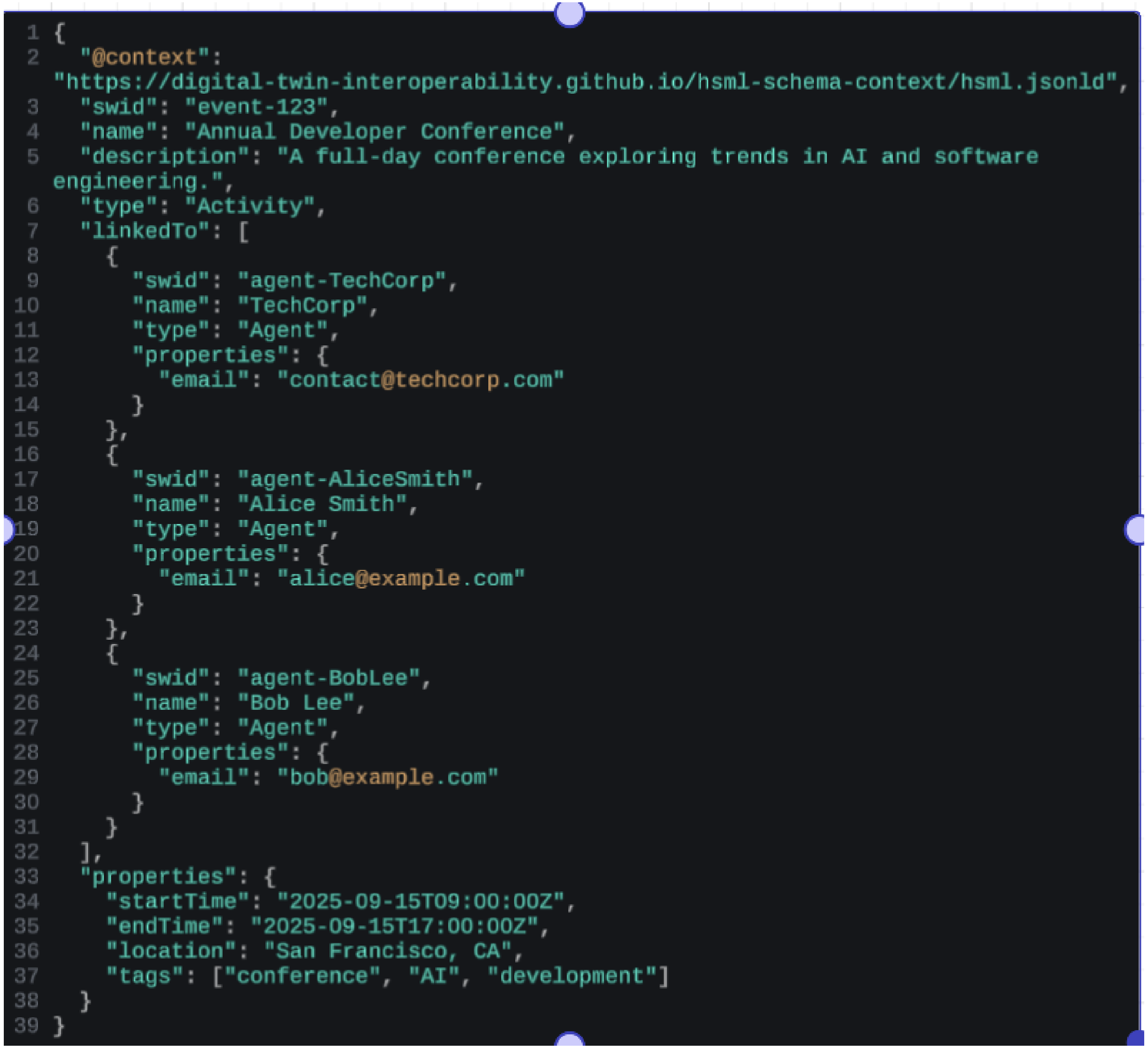


```json
{
  "id": "event-123",
  "title": "Annual Developer Conference",
  "organizer": {
    "name": "TechCorp",
    "email": "contact@techcorp.com"
  },
  "startTime": "2025-09-15T09:00:00Z",
  "endTime": "2025-09-15T17:00:00Z",
  "location": "San Francisco, CA",
  "attendees": [
    {"name": "Alice Smith", "email": "alice@example.com"},
    {"name": "Bob Lee", "email": "bob@example.com"}
  ],
  "description": "A full-day conference exploring trends in AI and software engineering.",
  "tags": ["conference", "AI", "development"]
}
```

OUTPUT

Key Mappings & Highlights

- **id → swid:** Ensures unique identifier within HSML.
- **title → name:** Human-readable label.
- **organizer & attendees → linkedTo:** Automatically creates Agent entries for each person/organization.
- **Remaining fields → properties:** Cleanly encapsulates additional data.



```
{
  "@context": "https://digital-twin-interoperability.github.io/hsml-schema-context/hsml.jsonld",
  "swid": "event-123",
  "name": "Annual Developer Conference",
  "description": "A full-day conference exploring trends in AI and software engineering.",
  "type": "Activity",
  "linkedTo": [
    {
      "swid": "agent-TechCorp",
      "name": "TechCorp",
      "type": "Agent",
      "properties": {
        "email": "contact@techcorp.com"
      }
    },
    {
      "swid": "agent-AliceSmith",
      "name": "Alice Smith",
      "type": "Agent",
      "properties": {
        "email": "alice@example.com"
      }
    },
    {
      "swid": "agent-BobLee",
      "name": "Bob Lee",
      "type": "Agent",
      "properties": {
        "email": "bob@example.com"
      }
    }
  ],
  "properties": {
    "startTime": "2025-09-15T09:00:00Z",
    "endTime": "2025-09-15T17:00:00Z",
    "location": "San Francisco, CA",
    "tags": ["conference", "AI", "development"]
  }
}
```

Value Demonstration

- **Accuracy:** Each field correctly mapped according to HSML schema.
- **Efficiency:** Eliminates manual mapping—conversion done in seconds.
- **Proof of Concept:** Demonstrates tool’s effectiveness and time savings in real use-case.

Integration

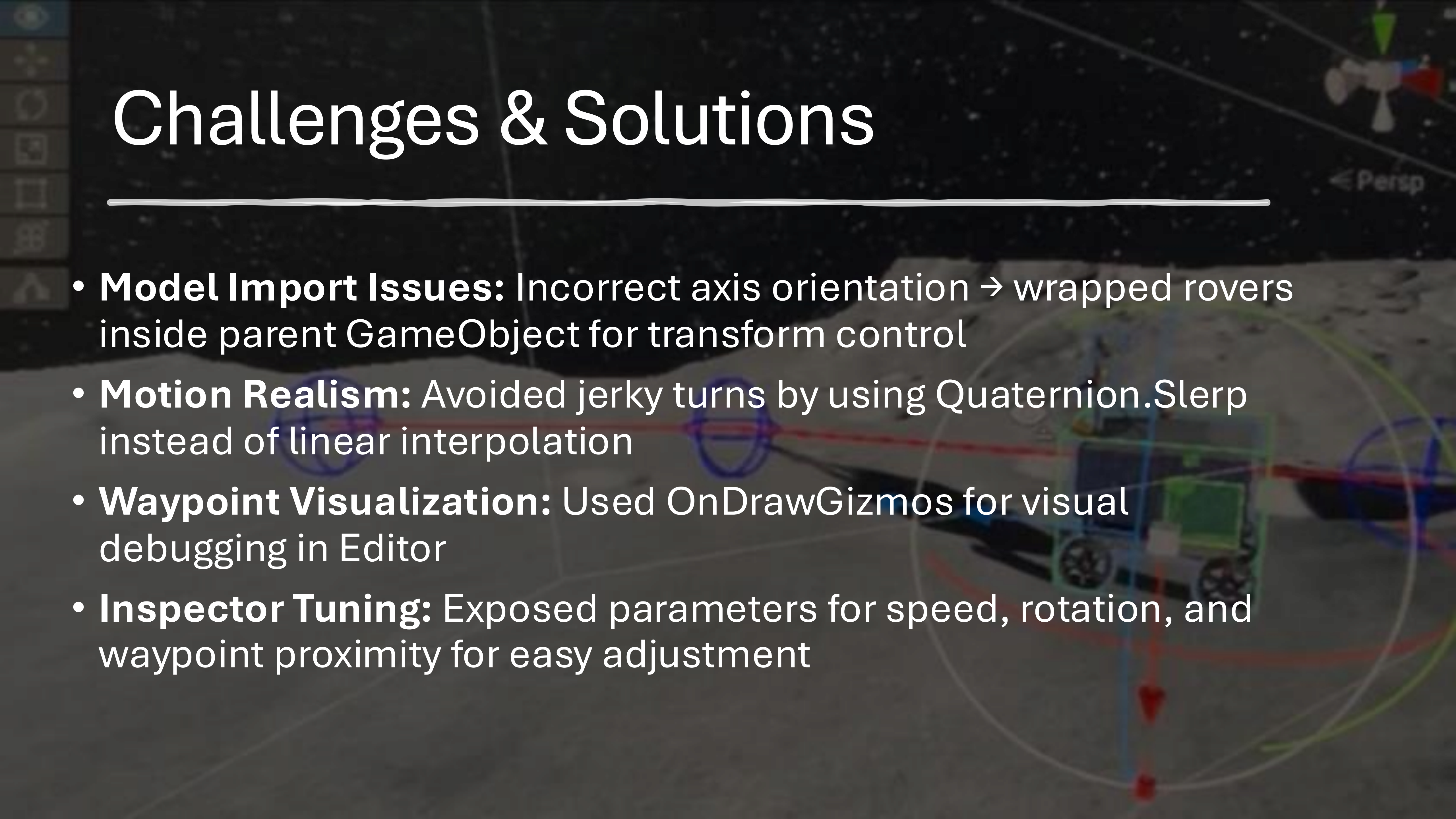
- **Agents A & B Interaction:** When **Agent A** needs to send data to **Agent B**, raw JSON is first transformed by our HSML conversion script.
- **HSML as Shared Language:** The converted HSML message ensures both agents speak the same schema, regardless of their native formats.

Autonomous Waypoint Navigation System in Unity

- **Created a modular system enabling rovers to follow user-defined waypoint paths**
- **Built with two main C# scripts: Waypoints.cs (path definition & visualization) and WaypointMover.cs (motion & rotation logic)**
- **Designed for reusable, inspector-configurable setup for any Unity object**
- **Supports smooth, realistic movement and rotation**

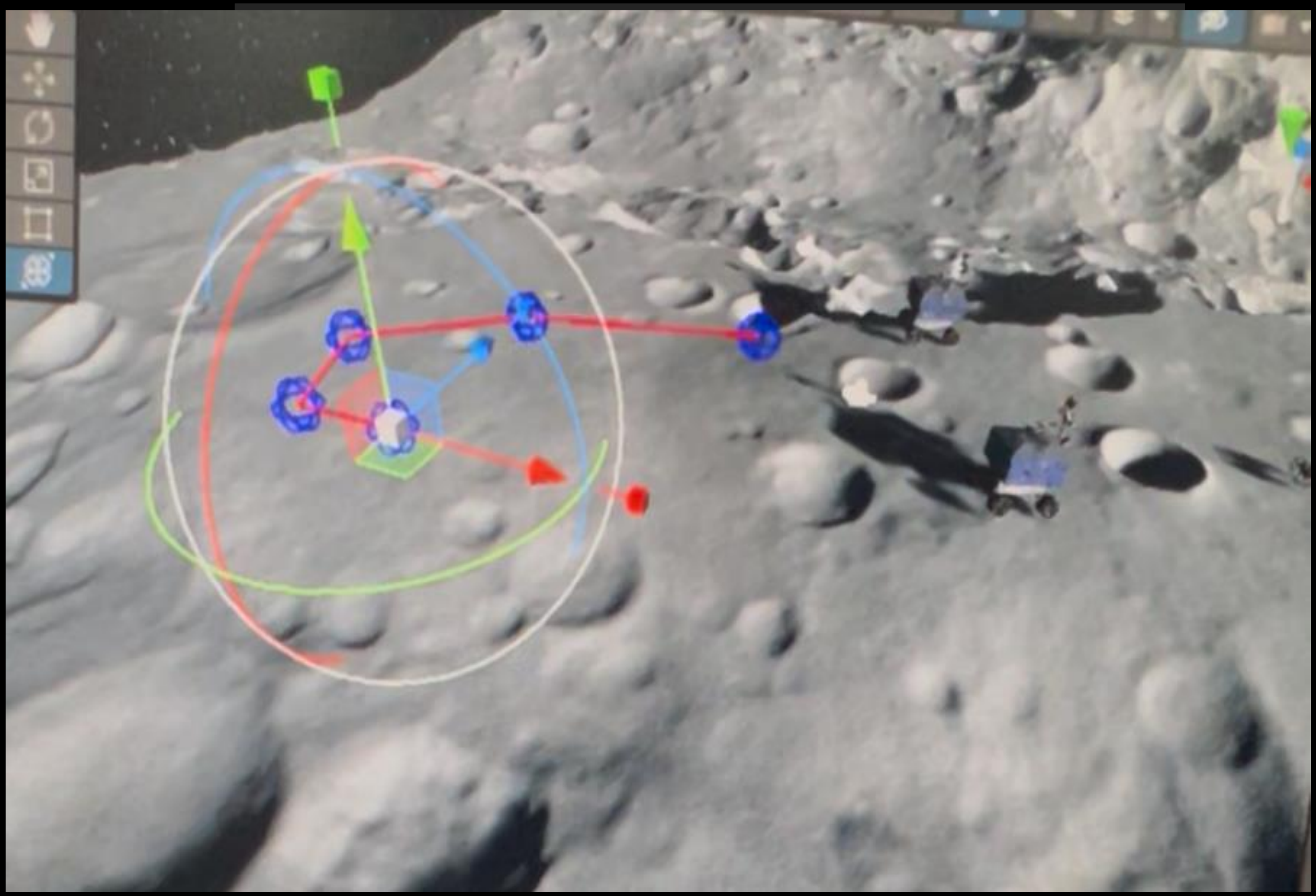
Challenges & Solutions

- **Model Import Issues:** Incorrect axis orientation → wrapped rovers inside parent GameObject for transform control
- **Motion Realism:** Avoided jerky turns by using Quaternion.Slerp instead of linear interpolation
- **Waypoint Visualization:** Used OnDrawGizmos for visual debugging in Editor
- **Inspector Tuning:** Exposed parameters for speed, rotation, and waypoint proximity for easy adjustment



How the Rover Moves & Turns

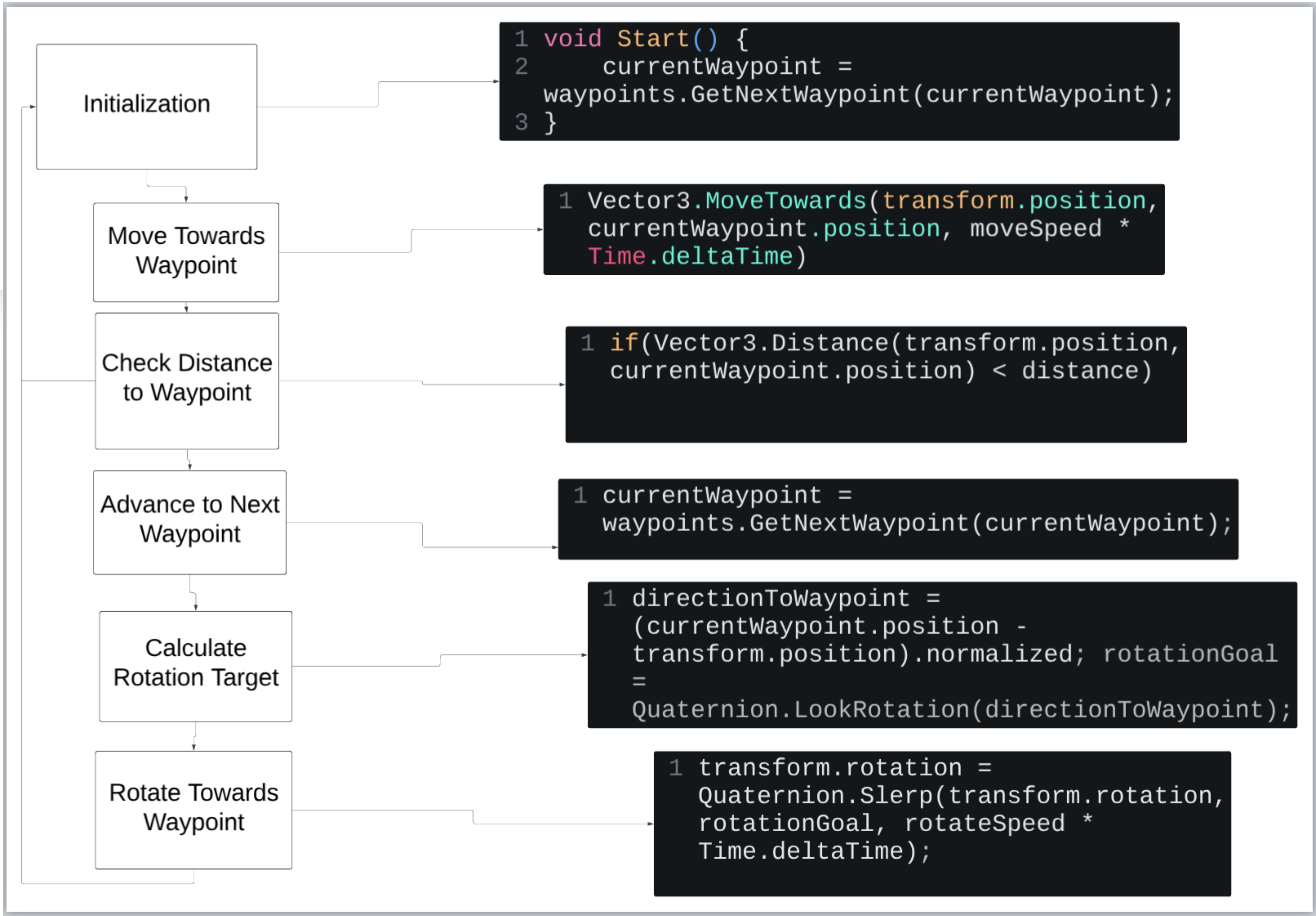
- Linear movement with Vector3.MoveTowards at constant speed
- Smooth heading adjustments using Quaternion.Slerp towards target waypoint
- Waypoint reached when within user-defined distance threshold, then advances to next waypoint
- Motion parameters adjustable in real-time through Inspector



Testing Strategy and Outcomes

- Tested within crater terrain Unity simulation environment
- Iterative debugging to fix axis errors, motion jitter, and waypoint threshold tuning
- System confirmed stable and reliable across all test runs
- Limitation: currently fixed speed between waypoints (no acceleration yet)

Waypoint Mover Script Flowchart



```
void Start() {
    currentWaypoint =
waypoints.GetNextWaypoint(currentWaypoint);
}
```

```
Vector3.MoveTowards(transform.position,
currentWaypoint.position, moveSpeed *
Time.deltaTime)
```

```
if(Vector3.Distance(transform.position,
currentWaypoint.position) < distance)
```

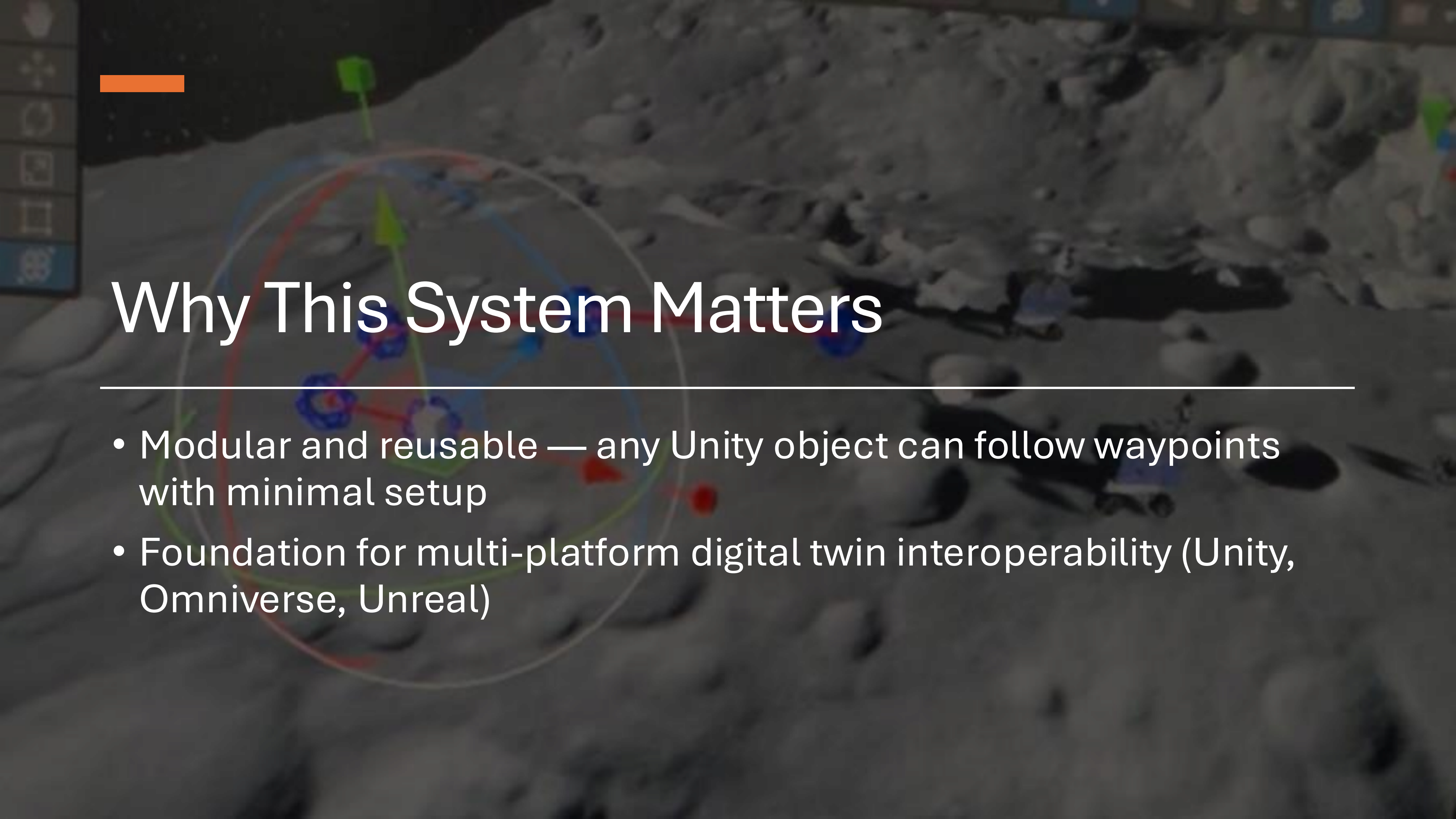
```
currentWaypoint =
waypoints.GetNextWaypoint(currentWaypoint);
```

```
directionToWaypoint =
(currentWaypoint.position -
transform.position).normalized; rotationGoal
=
Quaternion.LookRotation(directionToWaypoint);
```

```
transform.rotation =
Quaternion.Slerp(transform.rotation,
rotationGoal, rotateSpeed *
Time.deltaTime);
```

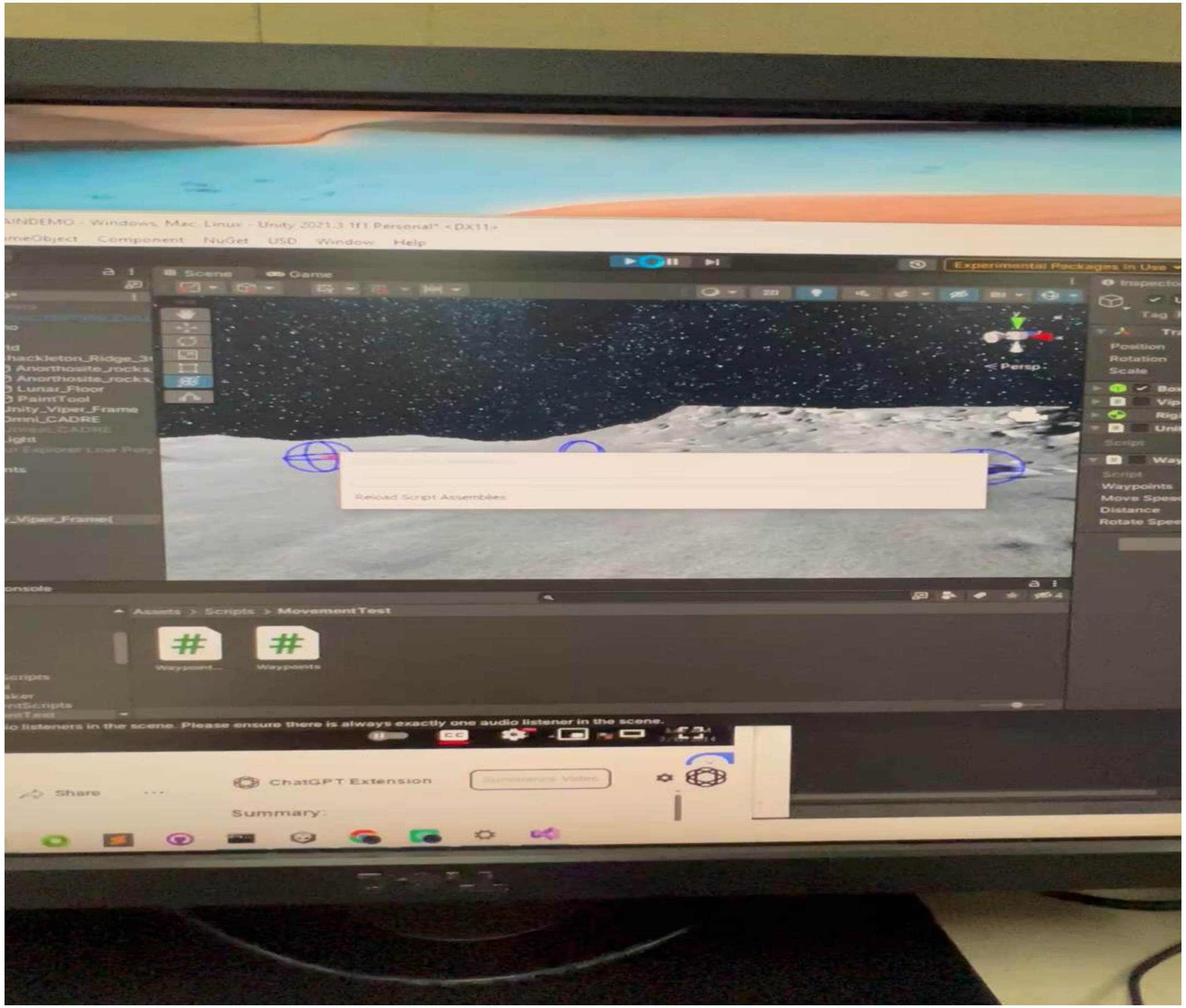
Why use [SerializeField] for these fields?

Field	Type & Default	Purpose / Description	Benefit for User
waypoints	Waypoints (Custom class)	Reference to the Waypoints object that holds the list of path points (waypoints) the rover will follow.	Allows the user to assign any waypoint path in the Unity Editor without modifying code.
moveSpeed	float (default: 2f)	Controls how fast the rover moves toward each waypoint.	Users can easily adjust rover speed in Unity's Inspector to fine-tune motion without touching the script.
distance	float (default: 1f)	Minimum distance to consider the current waypoint "reached" and advance to the next waypoint.	Lets users set how close the rover must get before moving on, controlling path precision from the Inspector.
rotateSpeed	float (default: 1f)	Controls how fast the rover rotates to face the next waypoint.	Users can adjust rotation smoothness and responsiveness easily via Inspector parameters.



Why This System Matters

- Modular and reusable — any Unity object can follow waypoints with minimal setup
- Foundation for multi-platform digital twin interoperability (Unity, Omniverse, Unreal)



AINDEMO - Windows, Mac, Linux - Unity 2021.3.1f1 Personal* <DX11>

Reload Script Assemblies

io listeners in the scene. Please ensure there is always exactly one audio listener in the scene.

Conclusions

- **HSML Conversion Tool:** Automated transformation of arbitrary JSON into standardized HSML format, enabling scalable, consistent data sharing across systems.
- **Waypoint Navigation System:** Created a modular Unity-based system for autonomous rover path-following, designed for reuse and integration into digital twin ecosystems.

Future Steps

- **Waypoint System:**
- Enable AI agents to dynamically generate and update waypoints based on path planning algorithms.
- Integrate with external AI control agents via socket communication for real-time autonomy.
- **HSML Conversion Tool:**
- Implement real-time data ingestion and conversion for live system integration.

Lessons Learned

- Modular code with Inspector settings makes testing and changes fast and easy.
- Keeping coordinate systems and object setups consistent is key for the simulation to work right.
- Using visualization tools like Gizmos helps a lot when finding and fixing bugs.
- Being close with mentors and asking lots of questions helps clarify what’s expected and ensures the project meets high standards.
- Real projects need a balance between technical accuracy and what’s practical to implement.

Acknowledgements

- A huge thank you to **Dr. Edward (Ed) Chow** and **Dr. Thomas Lu** for being so involved and supportive throughout my internship.
- Their guidance made this exciting experience possible, and I learned a lot in a new environment that I really enjoyed.
- Special thanks to the **program coordinators** for providing valuable resources and support during the SIRI meetings—it helped me stay on track and succeed.