THINKING IN WIT

WIT

An IDL (“Interface Definition Language”) for Wasm Components!

“Wasm Interface Types”

aka “Wit”

IDL: A SCHEMA LANGUAGE FOR PROGRAMS

- Protobufs
- Cap'n Proto
- Apache Thrift
- Mach IDL
- WebIDL

FAMILIAR DATATYPES

- `bool`, integer, floating-point, `string`
- list, record, variant, option

```
variant logo-command {
    turn(u16),
    forward(u32),
    pen(bool),
    color(tuple<u8, u8, u8>),
    speak(string),
}
```

AND ALSO...

FUNCTIONS

```
add: func(lhs: u32, rhs: u32) -> u32;

run-command: func(command: logo-command) -> result<_, error-code>

_
```

RESULTS

`result<some-value, error-code>`

This allow Wit to follow source-language conventions for error handling.

Rust:

```rust
pub fn test() -> Result<SomeValue, ErrorCode> {
    ...
}
```

Go:

```go
func MyApiFeatureTest() Result[MyApiFeatureSomeValue, MyApiFeatur
    ...
}
```

C#:

```csharp
public static unsafe
  global::MyWorldWorld.wit.imports.my.api.IFeature.SomeValue Call
{
  ...
  if error {
    throw new WitException<
      global::MyWorldWorld.wit.imports.my.api.IFeature.ErrorCode
    >(lifted.AsErr!, 0);
```

```
    }
}
```

RESOURCES

A resource is an entity that can be referred to by handles.

```
resource light-sensor {
    query: func() -> result<f32, error-code>;
}
```

HANDLES

Handles are “references” to resources, allowing users to pass them around and invoke methods.

CONCURRENCY TYPES

- `stream<T>`
- `future<T>`

MODELLING WITH WIT

```
resource my-laptop {
    power-button: func();

    press-key: func(k: key);
    touch-touchpad: func(where: touchpad-coordinates);

    screen: func() -> stream<frame>;

    get-usb-port: func(which: u8) -> option<usb-port>;
}

record touchpad-coordinates {
    x: f32,
    y: f32,
}
```

```
resource mouse {
     events: func() -> stream<usb-event>;

     left-click: func();
     right-click: func();
     move: func(where: mouse-coordinates);
}

resource usb-port {
     consume: func(events: stream<usb-event>);
}

enum key {
     // ...
}
```

FURTHER READING

https://component-model.bytecodealliance.org/design/wit.html