That

Joshua Furnish & Kai Austin

That

Mission: To put essential, core functionality of scripting at the fingertips of novice programmers as human-parsable (and grammatically correct) English

Proposed Solution

HyperTalk/HyperCard:

- Logic structure similar to Pascal
- Weak types
- Database capabilities

That:

- Modeled after HyperTalk
- Weak types
- Supports objects
- Interpreter interactions

Structure of Solution

Make an assignment:

Command (symbol) conjunction/preposition (expression)

Execute an expression:

Command (expression)

Using objects:

article + expression

ex: the (symbol) of (expression/object)

DEMO!

Interesting Challenges

Language Logic vs. Programming Logic

- Sense vs. Minimalism
- Translating grammar (i.e. “a” vs. “the”)
- Variability vs. Short cuts
  - Symmetrical Syntax

Symmetric Syntax

Normal symmetric syntax. Can mix up phrases, but mean the same thing:

```
>> make a car called nissan with speed of 50
>> make a car with speed of 50 called nissan
```

When symbols and expressions are not the same thing:

```
>> make a car with a speed of 50
>> make a car called Frank
    (i.e. define Frank as a car)
```

```
>> print “this” if TRUE
>> if TRUE, print “this”
```

Clash of Compromise

Add an “English” feature and break something else

```
>> define upit number as number + 1
>> print upit (upit 10)
```

Easy solution: Replace the parenthesis with “of”

```
>> print upit of upit 10
```

```
>> make a car called nissan with a speed of 50 and a size of 10
>> print (the speed of nissan) + (the size of nissan)
```

But then...

```
>> print of the speed of nissan + of the size of nissan
??
```