My experiences with Haskell as a person with Asperger's Syndrome

'Ow much 'askell would an aspie 'acker 'ack, if an aspie 'acker could 'ack 'askell?

Philippa Cowderoy

flippa@flippac.org

Outline

- What is Asperger's?
- General Coping Mechanisms
- Personal Background
- What's good about Haskell
- What I do with Haskell
- Community

Outline

- What is Asperger's?
- General Coping Mechanisms
- Personal Background
- What's good about Haskell
- What I do with Haskell
  - Architectural Astronautics
  - Linguistic Navel-gazing
- Community

What is Asperger's?

Asperger's Syndrome is an autistic spectrum condition occurring in individuals of normal or above intelligence. Issues and tendencies include:

- Impaired 'executive' function
- Problems with sensory perception and movement
- Social difficulties
- Systemic thinking (not always problematic!)

Impaired Executive Function

Problems with:

- Organisation
- Prioritisation
- Multi-tasking and shifting focus
- Motivation

All this frequently causes aspies problems looking after themselves, and can be rather draining.

Sensory Perception and Movement

- Senses can often become over- or under-sensitive
- Trouble filtering out background noise to listen to one thing
- Poor motor control - clumsiness!

In making it harder to do things, this also helps to wear me out faster. If I'm tired this can get even worse, and it can make me rather anxious.

Social Difficulties

- Not just ‘geeky’
- Trouble understanding people - what they mean, how they feel and why, why they act how they do
- Difficulty with body language
- Aspies are better at making themselves understood than understanding others

Compensating for this often has to be done ‘in software’, again tiring.

Consequences

The previous slides have used words like ‘draining’ and ‘tiring’. This means I get worn out faster and can do less in any given period of time than I might otherwise - further exacerbated by organisational problems!

If I over-exert myself it can turn into a vicious circle, getting to the stage where I have enough difficulty just making sure I’m eating. I have to guard my energy jealously.

General Coping Mechanisms

- Automation - Let something else do the work
- Optimisation - Make it all easier
- Routine - Maybe it does matter after all
- Introspection

General Coping Mechanisms

- Automation - Let something else do the work
- Optimisation - Make it all easier
- Routine - Maybe it does matter after all
  - Brainwashing yourself makes it all easier!
  - Self-automation
- Introspection - Understanding your head before hacking it!

Personal Background

- Early introduction to computers
- Special school
- Introduced to VB (version 1!) (but I survived!)
- Gaming leads to gamedev
- Gamedev leads to ~~the dark side~~C and C++
- But also to impatience and to scripting languages

Personal Background 2

Game development led to some interesting issues and balancing acts:

- Speed
  - x86 assembler (yuck!)
  - direct hardware access (maybe not so interesting)
- Managing multiple layers of abstraction
- Modifiability - how am I going to try out my game designs if I have to work with C++?

What’s good about Haskell

Enough with the background!

Manipulation and Abstraction

Takes advantage of mathematical skills I learned in school:

- Algebraic manipulation works
- Parameterising things is usually cheap or trivial, so generalising is easy
- Large changes can be accomplished by thinking about the transformations you need and applying them mechanically - taken step-by-step this is easy to get right and check

Everything explicit! No side-effects

- No getting bitten by unexpected side-effects...
- ...which also means fewer undocumented invariants
- Algebraic datatypes help to spot complete case analyses

Algebraic datatypes

- Pattern matching is easy - unlike the Visitor Pattern
- Ideal for language processing
- Using datatypes to structure functions

Architectural Astronautics

How can I organise myself?

...

I know, I'll build my own Personal Information Manager!

Flippi

- Barebones wiki, written with Haskell 98, old Text.CGI, Parsec
- Name suggested by Shae Erisson
- Evolved into a (thankfully never fully released) highly-configurable monster
- Galois used said monster for prototyping

Flippi plugin system

The big idea: allow plugins to modify the page IO operations (reads, writes, cataloguing...) by wrapping around existing code.

- ‘Base’ module becomes a record of operations, type PageIO
- Plugins supply a record of functions that transform each field in PageIO
- A configuration is a stack of plugins, with the ‘base’ module at the bottom

Plugin Problems

Problem: Plug-ins may need to do page IO of their own (eg update metadata), but mustn’t be allowed to circumvent plugins below them on the stack

Plugin Problems

Problem: Plug-ins may need to do page IO of their own (eg update metadata), but mustn’t be allowed to circumvent plugins below them on the stack

Solution: Tie a recursive knot, let plugins use the version with all plugins applied

What happened to Flippi?

- Released under a BSD license
- Used for prototyping by Shae and Galois
- Didn’t use it much myself in the end
- Never got enough UI polish - that’s hard work!

More PIM stuff

- What happens if you need multiple users? Upgrades and uptime? Can Haskell handle this?
- What happens if I want queries?
- Why do most of my PIM ideas turn into a lisp- or smalltalk-like system with a customised UI?...
- ...Turing Tarpit meets intentionally sloppy type discipline

Linguistic Navel-gazing

Haskell makes a great metalanguage!

It’s easy to write interpreters, compilers aren’t too horrific. Lazy evaluation and algebraic datatypes make it easy to translate from mathematical formalisms.

Haskell: PLT gateway drug

Haskell encouraged me to look at an increasing number of issues:

- Type systems
- Optimisation techniques
- So, so much semantics
- Algebraic methods
- Concurrency
- There’s always more

Monads as metaprogramming

Writing an interpreter for your own language counts as metaprogramming, so shouldn't implementing a monad too?

- Can be implemented as an interpreter, with combinators and computations building up an abstract syntax tree
- Lets us provide new semantics for »= and thus do

Monads: mutually-embedded DSLs?

Viewed as EDSLs, monads have their host language (here, Haskell) embedded in them via return - this is interesting, and not the same as merely using the host language to construct EDSL terms

Community

- General discussion - passing on ideas, spreading awareness
- Writing
- Projects - I may not be able to code so much, but I can mentor
- ...and I organised some event

Much I couldn’t or wouldn’t have done without encouragement from a number of people - a sense of community helps get things done

Things I learned from Haskell

Better programming, but that's not all:

- I learned to think about much bigger systems
- And thus how to understand more of life
- Haskell performance behaves more like many physical problems
- Maybe OO isn't such a silly model when you're interacting with many other systems

Sufficiently general programming is the study of general systems!

Conclusions

- Haskell amplifies clever thinking into good code
- We need to talk more about what it's like to use Haskell
- Strange circumstances don't necessarily stop you contributing
- Ideas can sometimes be useful far away from their original source