Если вы видите что-то необычное, просто сообщите мне.

# Часть 1: Ослабим страшный взгляд Haskell

Добро пожаловать в первую часть серии "Мозги Haskell"! Кроме описанных базовых понятий в прошлой серии статей, остается еще много работы, которая требует изучения нового! Эта серия статей затрагивает психологиеческие барьеры людей встречающие изучение нового языка. И дает советы для преодоления проблем.

Первая часть будет иметь совй взгляд на Haskell, в большом сообществе программистов. Мы посмотрим почему Haskell часто воспринимается как пугающий и трудный, и почему это не должно вас пугать.

Если вы бесстрашны и хотите попасть с корабля на бал, можно двигаться сразу ко второй статье. Там мы обсудим текущие процессы изучения языка. если вы хотите перейти сразу к изучению и написанию кода, то пожалуйста.

# Академический язык

Люди долго считали Haskell в основном языком исследования. Он построен на лябда вычислениях, возможно наипростейший, чистейший язык программирования. Это дает огромное количество возможностей связывания с отличными идеями в абстрактной математике, в первую очередь для студентов, профессоров и докторов. Эта связь настолько элегантна, что математические идеи могут быть легко представленны в Haskell.

Но эта связь имеет цену доступа. Важные идеи Haskell включают функторы, монады, категории и т.д. Они хороши, но только некоторые без математической степени имеют представление что значат эти понятия. Сравним эти понятия с другими языками: класс, итератор, цикл, шаблон. Эти гораздо понятнее, и языки используют в качестве

преимущества.

Отходя от этой терминологии, большой академический интерес это отличная вещь. Однако, на стороне производства, инструментарий не будет достаточный. Просто сложно обслуживать большие Haskell проекты. Как результат, компании не имеют большого интереса использовать его. Это значит, что нет особого влияния на академический баланс языка.

# Распространение знания.

Сетевые результаты Haskell академического первенства это перекошенная база знаний. В академии несколько человек проводят много времени на относительно маленьких проблемах. Учитывая другие академические поля, как вирусология. У вас есть некоторые эксперты которые понимаю вирусы на достаточно высоком уровне, и большинство не знают об этом ничего. Нет вирусологов-любителей. К сожалению, этот тип распространения знаний неблагоприятный для обучения новых людей теме.

Естественн, люди должны общаться с теми, кто знает больше их. Но правда в том, что они не хотят чтобы учителя тоже учились. Это сильно помогает в изучении если общаться с человеком который надавно касался этой темы. Скорей всего они помнят подводные камни и разочарования которые они встречали ранее, поэтому они смогут помочь вам избежать этого. Но когда распределение приходит в экстремум, нет среднего класса. Есть несколько человек которые могут обучить слушателей. В добавок не помня старых ошибок, эксперты используют сложную терминологию. Новые люди в теме могут чувствовать пугающее отчаяние.

# Перелом в производстве

Недостаток производственной работы, который обсуждался выше существенно способствует этому разрыву. Другие языки типа C++, имеет строгих академические последователей. Но после использования его компаниями в производстве, он не столкнулся с проблемой передачи знаний, которые имеет Haskell. Компании использующие C++ не

имеют выбора, кроме как обучать людей языку. Множество этих людей застряли в языке достаточно, чтобы обучить следующее поколение. Это создает более плавную кривую обучения.

Хорошие же новости для Haskell заключаются в том, что есть множество улучшенных инструментов за несколько последних лет. Это привнесло возрождение в язык. Множество компаний начали использовать его в производстве. Проходят больше встреч, больше людей пишут библиотеки, для большинства критических задач. Если это продолжится, Haskell надеемся достигнет переломного момента где распространение становится уже нормальным.

# Ключевая информация

Если вы один из тех кто заинтересован в изучении Haskell, или кто пытался изучить Haskell в прошлом, есть одна ведь которую нужно знать. В то время как абстрактная математика это излишество в повседневной жизни. Десятки языковых расширений должны выглядеть пугающими, но вы можете выбрать по одной.

На встрече Haskell eXchange 2016, Дон Стюарт из Standard Chartered начал разговоро о компаниях которые используют Haskell. Он объяснил, что они не часто используют что-то вне констукруций ванильного Haskell. Они просто им не нужны. Всё что вам нужно, скажем, линзы, вы можете получить без них.

Haskell отличается от большинства языков. Он ограничивает вас по своему. Но эти ограничения совершенно не являются тем на что они похоожи. Вы не можете исопльзовать их для цикла. Используйте рекурсии. Вы не можете изменять переменные. Поэтому создавайте новые используемые в выражениях. Просто каждый раз берете новую.

# Куда дальше?

Теперь зная, что Haskell не что-то страшное. Вы должны двигаться дальше. Зная подробнее о процессе изучения и нескиольких хитростях вы можете продолжить изучение дальше.

Revision #5

Created 2022-03-11 05:25:27 UTC by gasick

Updated 2022-08-21 08:06:00 UTC by gasick