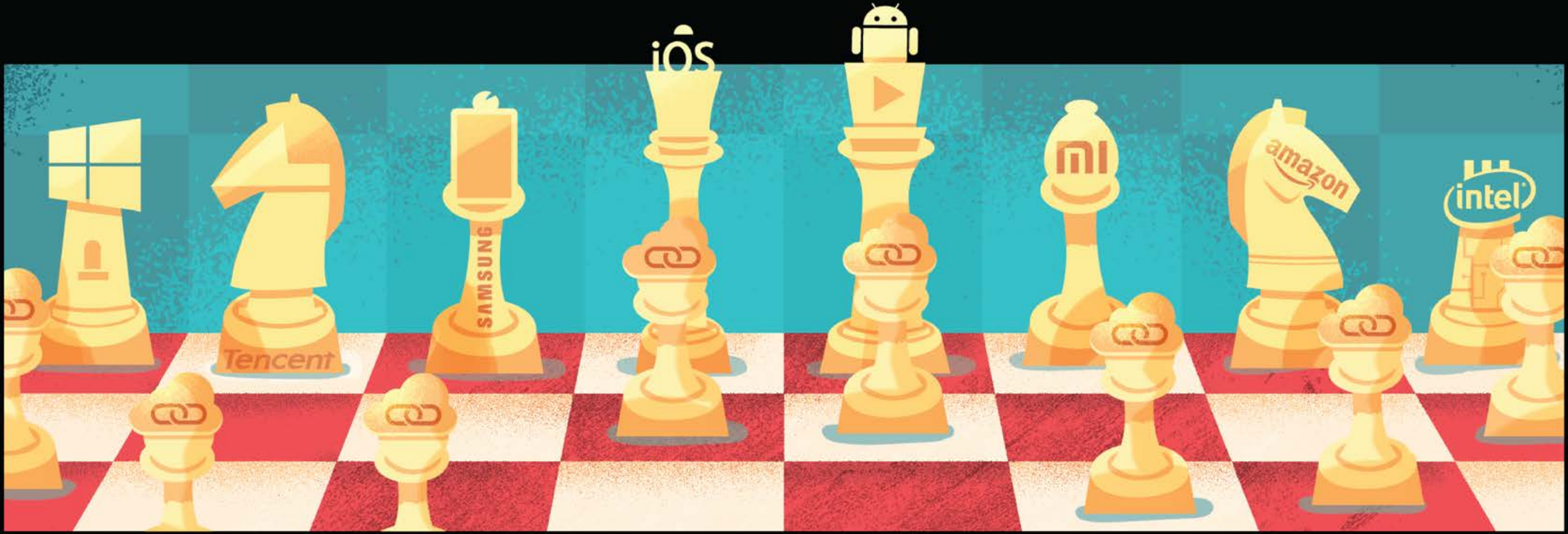
Created by: vision mobile

DEVELOPER ECONOMICS

# STATE OF THE DEVELOPER NATION Q3 2015

PUBLISHED july 2015

## Based on the largest developer survey of 13,000+ respondents



The 9th edition State of the Nation report now covers all the latest trends in mobile, desktop, IoT and cloud services development. We look at the most popular platforms, languages, vertical markets and hosting providers. Find out which types of development are bringing in the most revenue and which revenue models are succeeding. We also take a deep dive into mobile commerce, the biggest battleground in today's app economy.

vmob.me/DE3Q15

Marketing Partner: ubuntu
Supported by Canonical

# About VisionMobile ™

VisionMobile™ is the leading analyst company in the app economy and the developer ecosystem. Our surveys and app analytics track the changing landscape of mobile, IoT, desktop, and cloud developers.

Developer Economics is the largest, most global app developer research & engagement program reaching more than 13,000 developers in 149 countries. Our research program tracks developer experiences across platforms, revenues, apps, tools, APIs, segments and regions.

Our mantra: distilling market noise into market sense.

VisionMobile Ltd.
90 Long Acre, Covent Garden,
London WC2E 9RZ
+44 845 003 8742

www.visionmobile.com/blog
Follow us on twitter: @visionmobile

# Terms of re-use

**1. License Grant.**

Subject to the terms and conditions of this License, VisionMobile™ hereby grants you a worldwide, royalty-free, non-exclusive license to reproduce the Report or to incorporate parts of the Report (so long as this is no more than five pages) into one or more documents or publications.

**2. Restrictions.**

The license granted above is subject to and limited by the following restrictions. You must not distribute the Report on any website or publicly accessible Internet website (such as Dropbox or Slideshare) and you may distribute the Report only under the terms of this License. You may not sublicense the Report. You must keep intact all notices that refer to this License and to the disclaimer of warranties with every copy of the Report you distribute. If you incorporate parts of the Report (so long as this is no more than five pages) into an adaptation or collection, you must keep intact all copyright, trademark and confidentiality notices for the Report and provide attribution to VisionMobile™ in all distributions, reproductions, adaptations or incorporations which the Report is used (attribution requirement). You must not modify or alter the Report in any way, including providing translations of the Report.

**3. Representations, Warranties and Disclaimer**

VisionMobile™ believes the statements contained in this publication to be based upon information that we consider reliable, but we do not represent that it is accurate or complete and it should not be relied upon as such. Opinions expressed are current opinions as of the date appearing on this publication only and the information, including the opinions contained herein, are subject to change without notice. Use of this publication by any third party for whatever purpose should not and does not absolve such third party from using due diligence in verifying the publication's contents. VisionMobile disclaims all implied warranties, including, without limitation, warranties of merchantability or fitness for a particular purpose.

**4. Limitation on Liability.**

VisionMobile™, its affiliates and representatives shall have no liability for any direct, incidental, special, or consequential damages or lost profits, if any, suffered by any third party as a result of decisions made, or not made, or actions taken, or not taken, based on this publication.

**5. Termination**

This License and the rights granted hereunder will terminate automatically upon any breach by you of the terms of this License.

# Contents

**1.** The Global Developer Landscape
**2.** Choosing a Language is a regional, and financial, matter
**3.** Mobile Platforms: And then there were 2?
**4.** Chrome and Windows dominate the desktop mindset
**5.** Every Cloud has a lining, but they aren't all silver
**6.** Smart homes are still home to the smart money
**7.** Developer Revenues: There's more money in the cloud
**8.** Profitable developers are selling stuff, not eyeballs

# Also by VisionMobile

Find out more visionmobile.com/reports

The Industrial IoT Landscape 2015

IoT Developer and Platform Landscape 2015

Cross-Platform Tools 2015

Databoard

# ABOUT THE AUTHORS

## Bill Ray
Senior Analyst

Bill wrote his first mobile app in 1988, and has been failing to make money out of them ever since. He architected set-top boxes at Swisscom and Cable & Wireless, and was Head of Enabling Technology (responsible for on-device software) at UK mobile network O2. He then spent eight years as a journalist at tech publication The Register, before joining VisionMobile as a senior analyst.

You can reach Bill at:
bill@visionmobile.com
@bill4000

## Mark Wilcox
Senior Analyst

Mark has over 13 years of experience in mobile software across a variety of roles, however, he got his first computer at 4 years of age and wrote his first program at 7. He will probably always be a developer at heart. A growing interest in economics and business models has led him to work with VisionMobile as a Business Analyst, whilst still keeping his development skills up to date on development projects.

You can reach Mark at:
mark@visionmobile.com
@__MarkW__

## Christina Voskoglou
Director of Research and Operations

Christina leads the analyst team and oversees all VisionMobile research and data projects (big or small!), from design to methodology, to analysis and insights generation. She is also behind VisionMobile’s outcome-based developer segmentation model, as well as the Developer Economics reports and DataBoard subscription services. While at VisionMobile, Christina has led data analysis, survey design and methodology for the ongoing Developer
Economics research program, as well as several other primary research projects.

You can reach Christina at:
christina@visionmobile.com
@ChristinaVoskog

## About Developer Economics

Welcome to the State of the Developer Nation Q3 2015 report, the 9th edition of Developer Economics - the leading research program on mobile, desktop, IoT and cloud developers, tracking developer experiences across platforms, revenues, apps, languages, tools, APIs, segments and regions. The Developer Economics program investigates the latest trends in mobile development via semi annual developer surveys reaching up to 25,000 developers per year.

This is the 9th edition Developer Economics: State of the Developer Nation report and presents the key findings from the most global developer survey to date with over 13,000 respondents from 149 countries. This research report delves into the key mobile developer trends, as identified in our survey, and discusses demographics, platforms, languages, revenues, mobile commerce and more!

The report focuses on eight major themes – each comes with its own infographic:

1. The Global Developer Landscape - Developer age, gender imbalance and involvement in mobile, desktop, IoT and cloud development across regions
2. Choosing a language is a regional, and financial, matter - Developer education levels and revenues across languages
3. Mobile Platforms: And then there were 2? - The latest moves by the big platform players and shifts in developer mindshare
4. Chrome and Windows dominate the desktop mindset - Desktop platform choices and developer priorities
5. Every Cloud has a lining, but they aren't all silver - Cloud hosting platform mindshare and language specialization
6. Smart Homes are still home to the smart money - IoT developer preferences for vertical markets and hardware versus software development
7. Developer Revenues: There more money in the cloud - Developer revenues and revenue model choices across mobile, desktop, IoT and cloud
8. Profitable developers are selling stuff, not eyeballs - How the mobile app economy is shifting towards e-commerce as it matures

We hope you'll enjoy this report and find the insights useful!

If you have any questions or comments or are looking for additional data, you can get in touch at matos@visionmobile.com. You can also find an online version of our report at http://www.DeveloperEconomics.com/go

Bill, Mark, Christina, Matos, Alex, Andreas, Emilia, Dimitris, Vanessa, Sarah, Chris, Michael, Nick, Stijn, George and Dinos at VisionMobile.

@visionmobile

## Thank you

We'd like to thank everyone who helped us reach 13,000+ respondents for our survey, and create this report:

Our Marketing and Research Partners – Ubuntu, Intel, Microsoft and Amazon.

Our Leading Community and Media Partners, who are too many to number here – you know who you are!

# Companies who contributed to this research

## Supported by:

ubuntu
Supported by Canonical

## Research Partners

amazon.com

intel

Microsoft

## Lead Community Partners

ANDROID WEEKLY

BlackBerry

CSDN
全球最大中文IT社区

FICO

Powered by GameSalad

Google

InfoQ

jolla

OFFICINE ARDUINO

The Qt Company

W3C

## Media Partners

adduplex

appbackr

appindex
THE APP DEVELOPMENT MARKETPLACE

appnext

Apps4All
Developing your ideas!

APPTRACTOR

BongoHive

DEV LAB

GDG Kenya

HTML 5 APPS

ITMaster

mlab
southern africa

MOB4HIRE

MOBILEMONDAY

MTAM
Mobile Technology Association of Michigan

nailab
CHANGING KENYA, ONE STARTUP AT A TIME.

orange

pebble

SDR
superdevresources.com

SYMBIOTIC
design | develop | deploy

toptal

twilio

# KEY INSIGHTS

The 9th Developer Economics survey again scales up from previous studies, polling more than 13,000 developers from around the world to build a comprehensive model of the software industry. This year, for the first time, questions about desktop and cloud development were included, and we delved deep into development for the Internet of Things.

This State of the Developer Nation report highlights the most important findings from the survey: the trends in platform and language that are changing the industry, and how developers are making money with them. In this report we've summarized the key findings, and present some thoughts on how those trends will push developers in new directions, and how old revenue streams are being replaced by new ways to make a living.

## 10 Thoughts on the Developer Nation

- It's a very male-dominated nation. Only 6% of our respondents describe themselves as female, though the proportion was slightly better (10%) in North America.
- It's a population of young obsessives. 89% of developers are working across multiple areas, often filling their spare time with more software development.
- As a Nation it's close to the poverty line. More than half of mobile developers (51%), and well over half those in IoT (59%) aren't making a sustainable income (less than $500 a month).
- The Clouds here have a silver lining. Cloud developers are most likely to be doing well, 67% of them are generating decent revenue (more than $500 a month)
- Don't expect to see the Clouds out in public. The most popular Cloud development platform is decidedly private (targeted by 44% of developers), ahead of AWS (16) and Microsoft Azure (13%).
- Windows Phone doesn't come round so much these days. Interest in Microsoft's mobile platform has declined in the last six months (from 30 to 27%), while Android is still the most-targeted mobile platform (71%) and iOS remains popular.
- It's a multilingual Nation, with regional variations. In North America HTML5 is widely spoken (primary choice of 47%), while Asia leans towards more established languages such as Java and C.
- It's a Nation full of risk-taking explorers. More than a quarter (26%) of IoT developers don't even know who their eventual customers will be.
- The population is easily attracted by shiny new things. The Swift language is growing, despite the fact that 16% of developers working in Swift aren't making any revenue at all.
- It's an eternally optimistic Nation. 2% of desktop developers are targeting Google's Chrome OS, in the confidence that the day of the thin client is finally dawning.

The State of the Developer Nation is only scratching the surface of the data collected by VisionMobile, but within this report we can see a developer community expanding its horizons and constantly searching for new ways to do old things, and new ways to make them pay.

# Looking for the data behind this report?

## VISIONMOBILE DATABOARD SERVICE

Company size (number of employees)
Devices primarily targeted
Monthly revenues per app
Programming languages mindshare
Primary programming languages
Monthly app revenues by revenue model
Websites
mindshare

An annual subscription service that allows you to slice and dice thousands of app economy data points, via dashboards and interactive charts

Export graphs in multiple formats to use in your presentations

Download the aggregated data tables behind each graph to use in your analysis

Make sense of the data and leverage the expertise of our analyst team

vmob.me/DB

# 1 THE GLOBAL DEVELOPER LANDSCAPE

Our 9th Developer Economics survey received responses from more than 13,000 developers creating mobile apps, desktop apps, Internet of Things projects and backend services. From this broader base of respondents we have a view of most of the software development industry.

The global community of people involved in developing software for mobile devices, desktop computers, the Internet of Things and cloud services is fairly homogenous. They are predominantly young and painfully male-dominated everywhere. The level of gender imbalance, averages around 94% male globally, including non-developer roles like project managers and designers. Thanks to their relative youth, most developers also manage to find enough free time to be involved in several areas of development, via a hobby or side project in addition to their professional work. Amongst all this homogeneity, there's some interesting regional variation to explore.

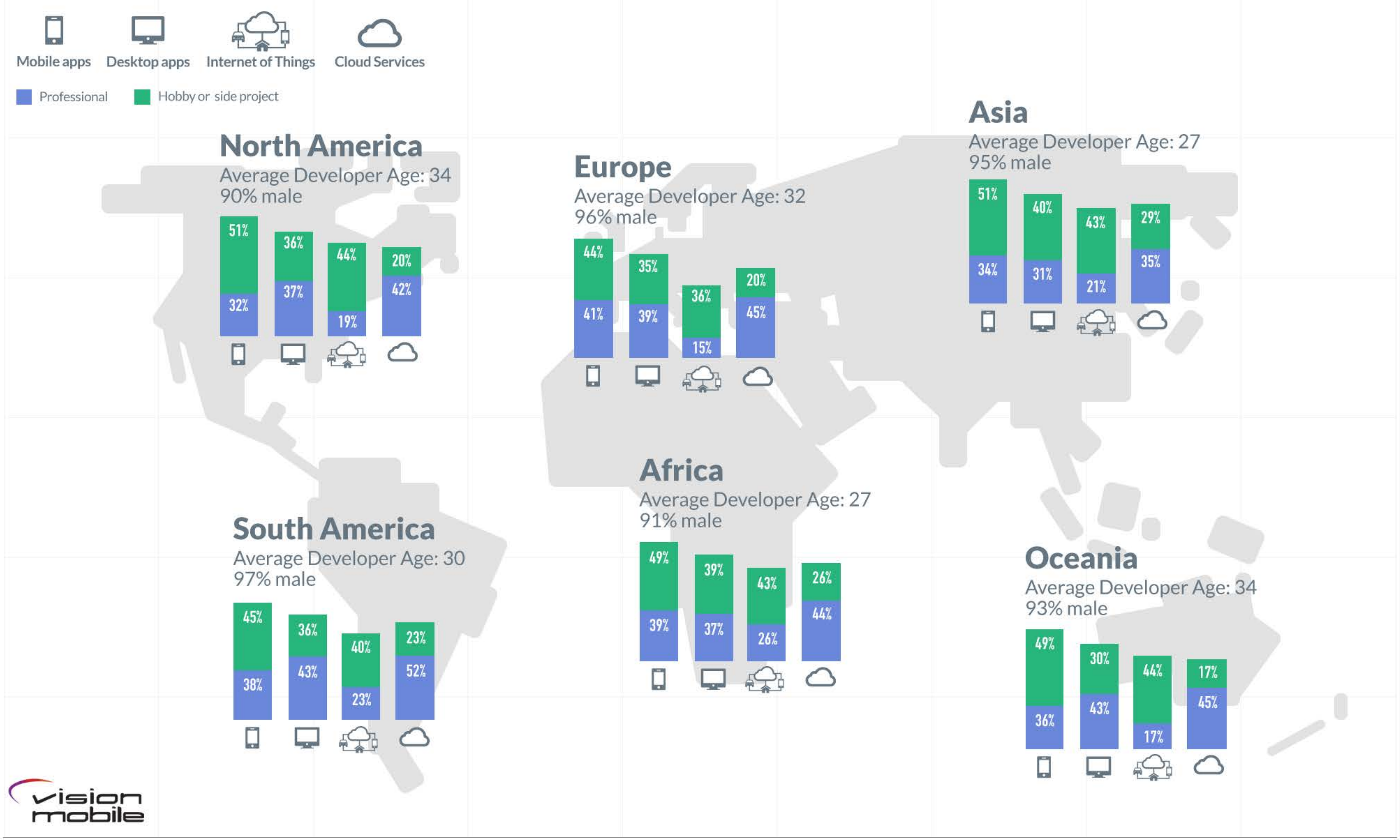
## The 'boys club' problem

It's clear that the software industry could benefit from much broader participation by women. Some early pioneers in computing, such as Ada Lovelace and Grace Hopper were women. As recently as the 1960s, computing was considered to be a career that was ideally suited to women and they dominated some areas. Unfortunately it somehow became less socially acceptable for girls and women to be nterested in computers and programming. This has to reverse if the industry is to both meet demand for developers and also to develop systems that really meet the needs of the entire population.

In North America there has been both strong criticism of the male-dominated culture in Silicon Valley startups and genuine efforts to get more women involved and welcomed in the industry. On balance they seem to be doing something right, as they lead the world in gender balance, with 10% females in the workforce. At the other end of the scale we have South America, with only 3% females. Unlike many countries (in Africa and the parts of the Middle East particularly) there is no serious issue with women having the opportunity to work in most of South America - it has a very high percentage of women in the workforce by global standards. Europe is the next worst with just 4% females. It's hard to find any excuse for this but solutions are needed. Recent moves to expose all children to programming at school from an early age in several European countries may help but will take many, many years to start making a difference.

# THE GLOBAL DEVELOPER LANDSCAPE

Asia and Africa have a younger technical workforce and are faster to exploit new areas like IoT



## North America

Average Developer Age: 34
90% male

## Europe

Average Developer Age: 32
96% male

## Asia

Average Developer Age: 27
95% male

## South America

Average Developer Age: 30
97% male

## Africa

Average Developer Age: 27
91% male

## Oceania

Average Developer Age: 34
93% male

Source: Developer Economics - State of the Developer Nation Q3 2015 | vmob.me/DE3Q15 |

## The experience problem

Software is a very young industry that has grown very rapidly. The natural consequence of this is a very young workforce. It’s true that the industry is evolving so rapidly it can be harder to keep up with the pace of technological change as you get older. However, that’s not a major reason we see so few older people in software development. 40 years ago there were almost no software professionals globally. Home computers only started appearing in any serious numbers a little over 30 years ago. To learn software development of any kind before that you generally had to be at a university with access to a computer that filled most of a room. You might have had to write programs by punching holes in a piece of card. There were so few opportunities to get started in the industry for people now in their 50s and 60s that there obviously aren’t many of them. Most of the really experienced leaders are in their 40s. The first generation that really grew up with computers are only in their 30s. As such it’s not at all surprising that the average age in the industry is just a little over 30. Computers were invented and personal computers commercialised first in English-speaking countries, so this probably explains why the average age in the industry is higher in those places.

> *Software is eating the world but it’s being written by a lot of people that are just figuring out how to do it as they go along.*

Most developers have come into the industry in fairly recent history. As we’ll see in Chapter 2, a large proportion of those have learned their computing at university. This creates a workforce with a very large number of junior developers for every experienced developer. Problems are created and mistakes are made on a very regular basis that could have been avoided with more experienced heads around. This problem is compounded by the relentless march forward of technology. Desktop computing just seemed to be starting to mature when focus rapidly shifted to the cloud and mobile. Mobile is still fairly new and developers are still figuring out the best ways to build software in that environment but already we have massive interest in the Internet of Things (IoT). A lot of general software experience can translate from one area to another but plenty of assumptions also need to be revisited. Software is eating the world but it’s being written by a lot of people that are just figuring out how to do it as they go along.

## The youth advantage

There’s one major advantage to such a young workforce. Most of them don’t yet have families and thus typically have plenty of free time to experiment. This can be seen in the enormous level of hobby and side project activity in the developer population. In most cases those with professional involvement in one area of development have hobby projects in others. The younger a developer is, the more likely they are to be involved in hobby projects. 89% of developers are involved in 2 or more areas of development. Most desktop developers got started in mobile through side projects and the same is now true for IoT. There are more than twice as many developers involved in IoT as a hobby or side project as professionally.

There are negative aspects to this too. Entry to new fields in software tends to be dependent on having time to experiment and build things on your own. This strongly favours those in the privileged position of having the wealth, health and leisure to pursue their interests. Beyond the gender bias, this also creates a fairly strong bias against people from less privileged backgrounds. However, if you have time and access to a computer, software is a relatively inexpensive hobby, so we see broadly similar levels of hobby activity (as a percentage of those involved at all) all over the world.

## Leapfrogging

An interesting observation from our survey data is that Africa, Asia and South America, with their younger developer populations are faster to shift towards new development areas. Europe and North America were first to adopt smartphones and tablets but the rest of the world has caught up in terms of professional mobile developers. Then with IoT, developing countries appear to be jumping ahead. Africa has the highest percentage of developers professionally involved of all regions. Europe has the lowest involvement in IoT - risk appetite is generally lower in Europe but this doesn't explain the relatively low hobby involvement. South America and Asia are both ahead of North America and Oceania. This is very promising for the future of IoT. It's unlikely that many people in Africa will need the IoT products being designed for consumers or large industries in North America. However, the same core technologies can be used to solve countless problems around the world.

# 2 CHOOSING A LANGUAGE IS A REGIONAL, AND FINANCIAL, MATTER

Development languages are still primarily driven by the target platform, though regional preferences are evident. Many languages are now cross-platform, translating common code into device-specific applications, giving developers more freedom to choose.

While some programming languages remain tied to a specific platform, there is an increasing range of alternatives available; not only allowing the creation of cross-platform application code but also attracting a new wave of developers keen to apply old language skills to new hardware. The days when Apple mandated the use of Objective C are long gone – the policy of refusing to sell applications developed in any other language was only intended to stall deployment of Adobe AIR, and has served its purpose.

Apple now offers Swift alongside Objective C, for iOS development, but toolchains are available for a wide range of languages including JavaScript, C#, Ruby and Python, as well as more obscure lingos and variants.

Platform owners hoping to attract developers need to support a range of languages, but maintaining libraries and APIs can prove expensive and dropping a language is politically difficult. Google's approach with Android limits most APIs to applications using the Dalvik virtual machine. Android applications wanting to use those APIs generally compile down to Dalvik bytecode for deployment, regardless of the development language, or bridge frequently to the Dalvik virtual machine at runtime. This adds complexity to the use of alternative languages, making Java the easiest option, without Google needing to impose restrictions.

Regional differences, such as the North American preference for HTML5 discussed below, also exist. These are based on cultural history and economic imperatives, as developers are understandably reluctant to invest in learning a new language when they can instead apply their existing experience.

> *Platform owners hoping to attract developers need to support a range of languages, but maintaining libraries and APIs can prove expensive and dropping a language is politically difficult.*

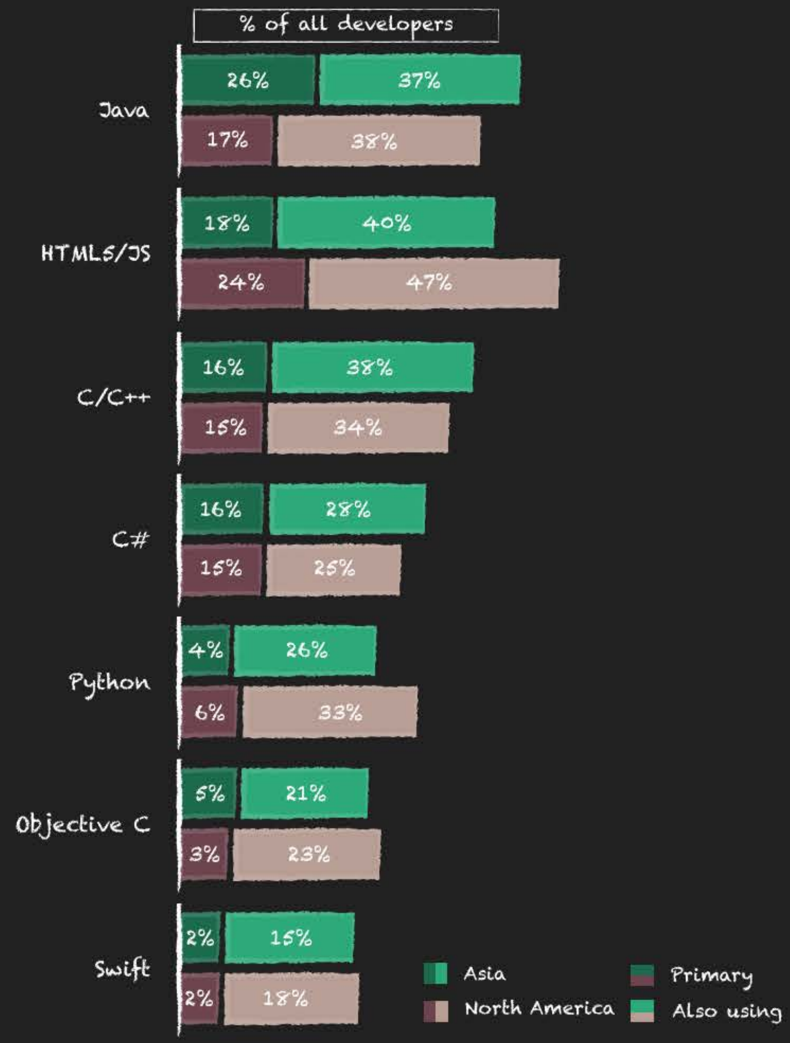
## Polyglot programmers

Many programmers are comfortable in multiple languages, and in some instances the differences are small enough to make skills transferable, but as we noted in our last report (State of the Developer Nation Q1 2015), the rise of Apple's Swift language has not been at the cost of Objective C developers – Apple has attracted a new wave of programmers who were, perhaps, intimidated by the perceived complexity of Objective C.

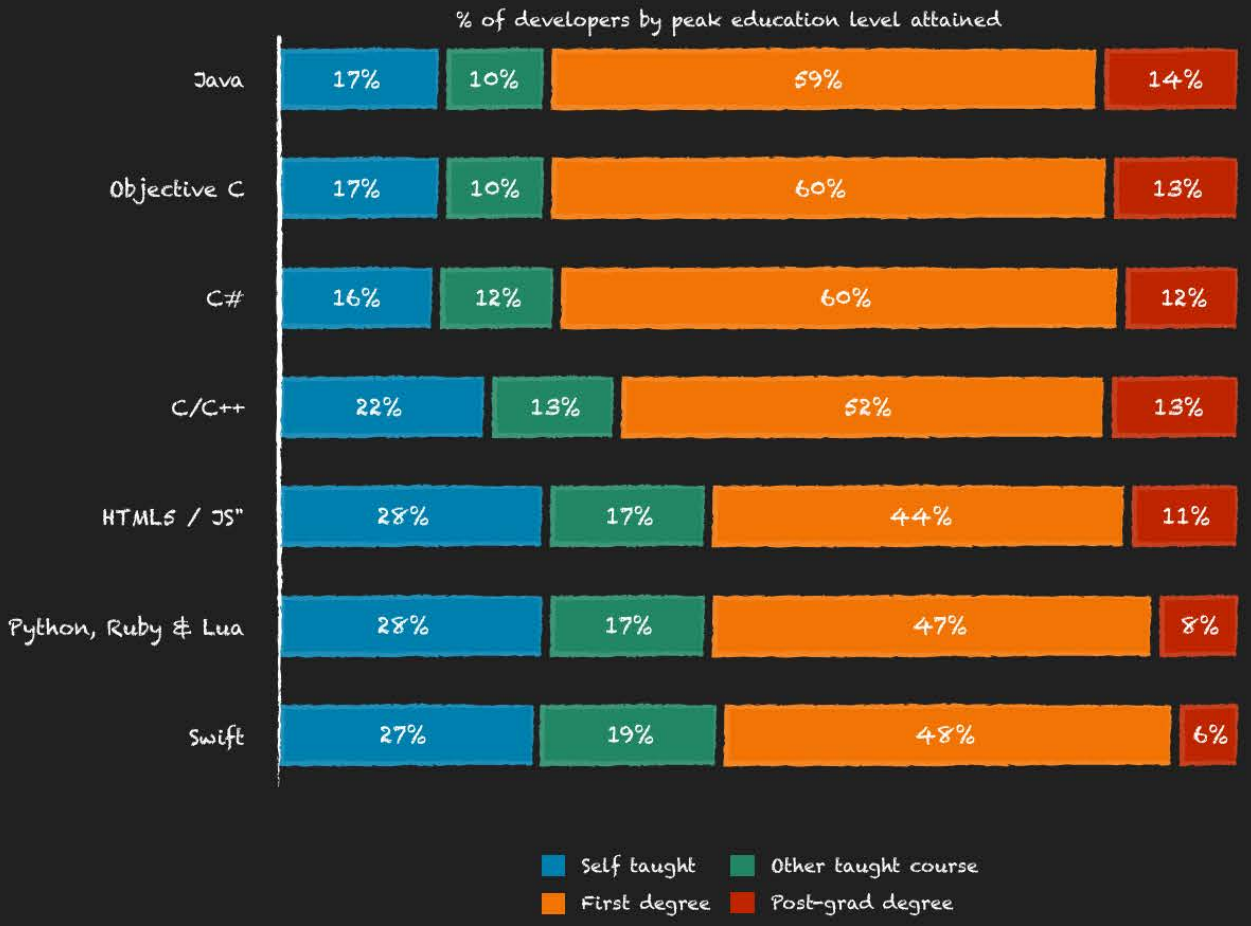
# OLD-SCHOOL LANGUAGES ATTRACT THE MOST OLD SCHOLARS

More than 65% of Java, Objective C, C# and C/C++ developers have a relevant university degree

Old-School languages are favoured more in Asia



More self taught developers favour newer languages



vision mobile

Source: Developer Economics - State of the Developer Nation Q3 2015 | vmob.me/DE3Q15 | Licensed under CC BY ND | Copyright VisionMobile

The arrival of Microsoft, and Windows Phone, should not be ignored as a driver towards greater flexibility in development. Microsoft has long supported a wide range of languages for desktop development, from Visual Basic to C++, and has more recently extended this flexibility on to its mobile platforms. Pushing developers towards a single language; as Apple and Google did with Objective C and Java respectively, was very effective in building a community, but now risks becoming a limitation not shared with the competition.

In IoT development there are more options, as the plethora of platforms have yet to coalesce into standards. Developers are using the languages they know, a trend shared with those working on IoT apps, though the latter may be forced into a decision by the need to interoperate with existing code.

## Regional preferences

Regional variations in the choice of development language can sometimes be attributed to the dominance of specific hardware platforms, but the numbers also show regional preferences beyond that – factors which can more-easily be attributed to market conditions and economic realities, rather than dictums from platform owners.

Java remains very popular in South Asia, where mobile development has long been aimed at "feature phones", devices which support J2ME (Java 2, Micro Edition) but cannot be considered smartphones. Android, and its derivatives, have also been very successful in the region. India's Jivi is selling Android handsets for a shade over $30, replacing the features phones while maintaining Java as the preferred development language.

Equally interesting is the support for scripted languages, and HTML5, in North America where Hypertext is the primary platform for 47% of developers. While developers elsewhere are sticking with Objective C and C# (the primary platform for 38% of Asian developers) the Americans are looking at cross-platform and rapid-development solutions, a trend which is mirrored in Western Europe and Israel.

> *The Americans are looking at cross-platform and rapid-development solutions.*

HTML5 development is easier with reliable connectivity, despite the addition of persistence APIs, web pages still work better when connected. Scripted solutions are also harder to optimise, so their popularity may reflect the greater processing power available to North American consumers. Scripting languages are generally faster to work with, and slower to execute, but if the end users have the latest flagship hardware then the execution speed is less critical.

North America also has a legacy population of web developers, highly skilled in HTML and JavaScript and keen to apply that experience into the mobile marketplace.

## Some languages are more equal than others

Developers are getting very creative when it comes to generating revenue, and most are keen to pick a language which keeps a wide range of options open. Niche languages are most popular amongst those less interested in making money (more than a third of Python developers fall into this category), while Objective C developers top the table when it comes to contracting, as 35% of them expect someone else to pay for their time.

ActionScript is an interesting anomaly in the figures, as mobile developers using Adobe's language are focused on advertising and downloads as a revenue stream, while developers working in other languages (notably Java) are migrating towards subscription and value-added models. It's worth noting that 27% of Objective C programmers hope to make money from advertising, while only 20%

of those working in Swift have similar aspirations, demonstrating how a new wave of developers is adopting new revenue models.

In the IoT developer community it is harder to see trends emerging, as the nascent industry is still working out how to pay for itself. The majority of IoT developers, across all languages, are still at the exploration stage where revenue isn't an issue.

In PC development the revenue sources are very evenly distributed, reflecting a mature industry with well-established channels.

## Mobile revenue still evading the Swift

Amongst mobile developers there seems little correlation between choice of language and total revenue generated. Revenue is spread across the range fairly evenly across the languages.

Those using visual development for mobile applications do tend to yield less directly, with 20% of reporting zero revenue, but that's likely because those applications have alternative sources of funding – promotional applications, or those linked to a physical service such as cinema listings or train times.

16% of those working in Swift are still not generating revenue directly, which is interesting when contrasted with the 4% of those programming in Objective C. It's possible that this reflects the applications for which the languages are being used – with Swift creating informational and promotional applications, while Objective C is used for revenue-generating games and apps – but it is more likely to reflect the immaturity of Swift, as developers are still creating applications which will be profitable over time. We should see that change over the next six months, as more Swift applications come to market.

In IoT things are more polarised, with C/C++ developers taking the lion's share of the money available. That reflects the embedded nature of IoT device development, which often requires the use of low-level languages and specialist skills to reduce power consumption and keep processor costs low. Software running inside a light switch or a door lock is resource-constrained and software able to run within such constraints commands a premium rate.

HTML5 and JavaScript both have a significant bump towards the high end, with more than a third of each reporting monthly income of more than $100K ($200K for JavaScript) from IoT development. While embedded development is essential for the 'things', there are also interface screens and server-side processing logic where speed of development is more important than speed of execution, and this is where the web-based technologies excel.

## Education, Education, and Education

Developers are typically self-starters, and that is reflected in the predominance of self-teaching across all the languages. Around two thirds of developers claim some sort of self-education, though many have followed that up with degrees and post-graduate studies, and it is clear that some languages lend themselves to formal education.

Java programmers have the biggest percentage of degrees (73%), perhaps reflecting the way that Oracle (and, previously, Sun Microsystems) pushed Java as an ideal learning language. The availability of free developer kits certainly encouraged many universities towards Java in the early days, but the scalability of the language continues to recommend it for educational use.

> *Java programmers have the biggest percentage of degrees*

Objective C and C# have very similarly-educated developers to Java. Meanwhile, C/C++ also boasts a slightly lower number of degree-holders (65%), and almost ties with Java for postgraduate qualifications. The lower proportion of degrees for C/C++ developer is mostly explained but more self-taught games developers using the language.

Strongest amongst the self-taught are the niche languages;  Ruby, Lua and Python (average 28% self-taught), alongside Swift (27%) which is new enough that anyone using it will have to have taught themselves to some degree. Visual Development tools also show a high proportion of self-learners, demonstrating the advantages of having a low barrier to entry.

Amongst alternative education routes on-the-job training dominates, with programmers expected to improve their skills through experience and collaboration. Classroom training courses are clearly popular with the higher-level languages, including scripting languages and HTML5, while Massively Open Online Courses are providing education to those studying Python and Objective C, but also Swift, showing that the provision of such courses is an important tool for platform owners hoping to attract developers.

# 3 MOBILE PLATFORMS: AND THEN THERE WERE 2?

In our Q1 2015 State of the Developer Nation report, we highlighted the stalemate in the mobile platform wars. Apple was increasingly coming to dominate the high end of the market with iOS, leaving Android with everything else.

Microsoft's Windows Phone ecosystem was still lacking sufficient scale to compete, but slowly growing developer interest. Six months later, Apple's dominance of the high-end has only increased as the only premium Android manufacturer with scale, Samsung, loses more ground. However, Android still owns the rest of the market and Microsoft seems to have finally thrown in the towel. Meanwhile the mobile web continues to make a steady comeback, although as a supporting platform, rather than a primary one, in most cases.

## Apple's advantage

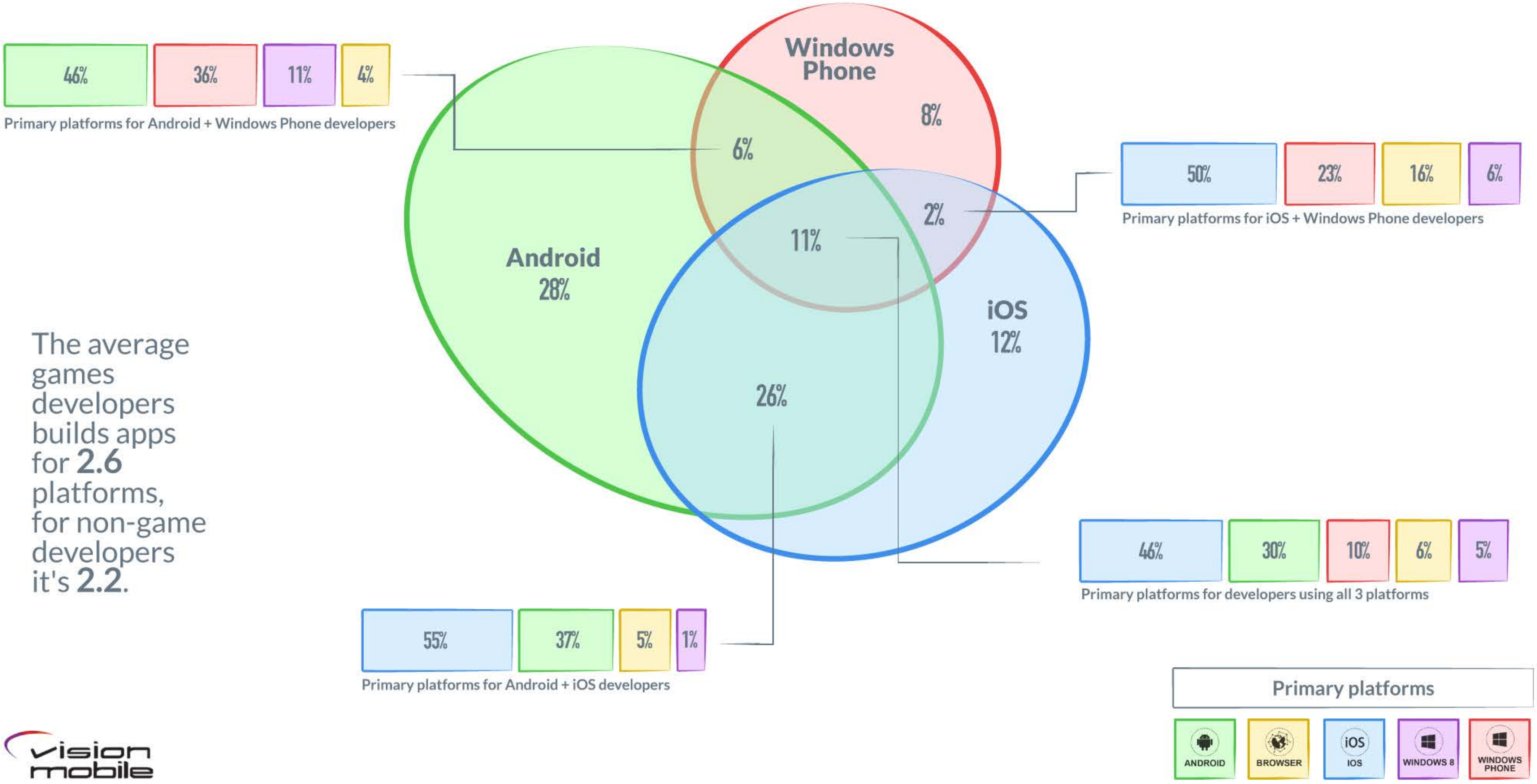
Apple combines premium materials and manufacturing processes with a differentiated software platform, creating devices which are distinctive and easily recognised. They only make premium products, so owning one is a signal of status and wealth. With the focus resolutely on usability, at the expense of customisability, the brand loyalty is maintained and extended into the entire experience. A majority of those who can afford an iPhone want one. Apple then grants developers exclusive access to the market of users with the highest disposable incomes. This creates a positive-feedback cycle where developers create higher quality apps for Apple's platforms, competing for the attention of those users and attracting more of them to the platform.

Many observers of the market suggest that the iPhone will eventually over-serve the needs of users. Cheaper modular alternatives, based on Android, will be good enough and people will not be willing to pay the premium for Apple's products. A strong counter-argument to this is the Mac range. Apple's desktop and laptop offerings have been growing market share in a declining market. Where most users can accomplish their work on an aging laptop, costing as little as $500 new, an increasing proportion are being convinced to buy a premium option from Apple for $1000+. In many cases it's the experience they've had with an iPhone that makes them do this, the mobile device acting as an introduction to the Apple ecosystem. At the same time, Apple has managed to increase the average selling price of an iPhone, demonstrating the power of the platform. It certainly doesn't appear as if the iPhone is going to lose many high-end customers to Android any time soon, so the iPhone developer base is going to keep producing new apps.

# MOBILE PLATFORMS: NOT IOS OR ANDROID BUT BOTH

## 37% of all mobile developers target both iOS and Android, only 7% target none of the top 3 platforms

% of all mobile developers using each combination of the top 3 platforms



The average games developers builds apps for **2.6** platforms, for non-game developers it's **2.2**.

vision mobile

Source: Developer Economics - State of the Developer Nation Q3 2015 | vmob.me/DE3Q15 | Licensed under CC BY ND | Copyright VisionMobile

## Android is essential

When the value of the app economy was dominated by app store revenues, Apple's user demographic advantage was overwhelming. Now that the app economy is maturing and moving towards more complex revenue models, such as subscriptions and e-Commerce, Android's reach is too large to ignore. Apple users may spend more per transaction, but the quantity of users buying through their Android devices will more than compensate for this over the next few years. It seems likely that for the most popular apps of the future, iOS-only, or even iOS-first, may no longer make sense. Currently 37% of all mobile developers target both iOS and Android, and we expect their numbers to grow. If we exclude the Hobbyists and Explorers in our segmentation model, counting only those developers building apps for a living, then 44% are already supporting both platforms. Android remains by far the most popular platform overall; targeted by 71% of all mobile developers with 28% only using Android from the top 3 platforms (Android, iOS and Windows Phone). However, where both iOS and Android are used, iOS takes priority by a significant margin. This still makes sense: even if the majority of your customers are on one platform, you still prioritise the premium platform where your best customers are to be found.

## Windows Phone is dead, long live Windows 10

Microsoft has been spending heavily on Windows Phone for several years in an attempt to catch up with the two mobile ecosystem leaders. In our survey we saw a positive sign for the platform with a slightly better mix of professional versus hobbyist activity. Another potentially positive sign for Microsoft is that 44% of mobile developers said they were planning to adopt Windows 10 versus a historical 28% for Windows 8 at the same point in the Windows 8.1 release cycle. Then again, intent to adopt Windows Phone peaked at 57% in 2012. Unfortunately we also saw a drop in the overall fraction of developers targeting the platform; falling from 30% in Q1 to 27%, and reversing most of the gains of the previous 12 months. This context makes Satya Nadella's decision to scale back Microsoft's mobile ambitions seem very wise. Microsoft has effectively written off the entire value of Nokia Devices, the business purchased in 2013 for $7.2bn, and are making most of the remaining staff that came with the acquisition redundant. They are streamlining their portfolio to focus on fewer devices. This signals that they don't expect to ever make a significant profit selling smartphones - the Lumia range will probably play a role similar to Nexus devices at Google, as flagships and reference models promoting the platform. Deep involvement in real products gives OS engineers a much better idea of what's required from the platform, and how it should be optimised. Sales volume is not the target, so marketing spend is significantly reduced and device subsidies will not be required to stimulate demand . This obviously means that where Microsoft was by far the biggest seller of Windows Phone devices, overall sales volume and market share will fall.

> *We'd expect to see developers migrating away from Windows Phone, and Windows 8, to Windows 10, and this will still happen, but the final total is likely to be well below Windows Phone's current level of interest.*

Developers are likely to abandon Windows Phone/Mobile-related efforts and switch their focus to one or both of the leading platforms. Xamarin will probably be a major beneficiary, helping developers bring their C# code bases to iOS and Android. This abandonment will be hard to track accurately because Windows Phone is being merged into Windows 10, Microsoft's common platform available from July 29th 2015. We'd expect to see developers migrating away from Windows Phone, and Windows 8, to Windows 10, and this will still happen, but the final total is likely to be well below Windows Phone's current level of interest. Although Microsoft is showing positive sales momentum for their Surface tablets, the tablet market overall is slowing significantly. Users are not seeing the need to upgrade tablets with the same frequency as their smartphones, particularly as smartphone screen sizes have grown. Microsoft's future hopes in mobile will depend on producing innovative new product lines.

## The mobile browser comeback

A year ago we saw developer interest in the mobile browser, as a platform hit an all-time low - just 15% of mobile developers were targeting the browser at all. Now 26% of developers are targeting the mobile browser, putting it in a very close 4th place behind Windows Phone. However, still only 7% of mobile developers consider the browser to be their primary platform. A mobile website or web app is now increasingly seen as a necessary addition to a native app. Users are becoming more reluctant to download new apps just to try them out, and developers can't rely on app stores for discovery. A presence on the mobile web enables developers to provide a better idea of the product or service on offer - more interactive than an app store listing. Having a desktop-only website that points mobile visitors to the app stores is increasingly seen as unacceptable.

Both Android and iOS have been adding features to facilitate the interaction between web and native apps. However, while Google push ahead with new browser features designed to close the gap between web apps and native apps, Apple has no incentive to do so. Desktop Safari has push notifications, whilst Mobile Safari does not. Chrome for Android has Service Workers; enabling background processing and better offline support while Apple appears in no hurry to implement this draft standard. That's causing frustration to the web-first diehards, who are desperate for a competitive platform, but in most cases the new browser features that can't be used on iOS are features that can't be used, at all. If the web can make a real comeback, and avoid being pushed into the reduced role of supporting actor, then Apple holds the keys.

# CROSS-PLATFORM TOOLS 2015

## What do 8,000 developers and 185,000 apps reveal about the future of the market?

An analysis of the CPT market, based on survey data from 8,000 app developers and analysis of 185,000 apps in the Google Play Store.

Tracking market leaders Cordova, Xamarin, Unity, Qt and up-and-coming challengers, including Facebook's React Native, and Telerik's NativeScript.

vmob.me/CPT15

# 4 CHROME AND WINDOWS DOMINATE THE DESKTOP MINDSET

It's not a surprise to see Microsoft Windows continuing to dominate desktop computing, and we can see that clearly reflected in the platforms targeted by developers.

A solid third of desktop developers are aiming squarely at old versions of Windows, though with Microsoft's platform being (reasonably) backwards compatible that can be combined with the 8% who are looking to deploy on more-recent versions to make Microsoft the dominant platform.

But only just, as the number of developers creating apps designed for web browsers continues to creep up. We can split Windows between old (classic) and new (modern) - which makes sense given the Windows Store is primarily aimed at Windows 8, which Microsoft considers the first "modern" windows - then the browser tops every platform as the most-popular target for desktop developers.

Browsers are supposed to be standards-compliant, but despite occasional industry initiatives they remain fragmented in their more-advanced capabilities. That leaves developers to target specific browsers, or at least a selection of the most popular, which in turn means Chrome and Firefox. That's for public apps, but in the enterprise things are slightly different - a good many enterprise users are stuck with Internet Explorer 8 for compatibility with existing apps, so developers are continuing to target browsers which would be considered redundant outside corporate verticals.

## Will a Spartan alternative attract developer minds?

Microsoft has just launched a new browser, called Edge and bundled with Windows 10 it comes without the legacy of backwards compatibility. Users of Windows 10 will still be able to use IE, to maintain support for deployed applications, but Microsoft is hopeful that Edge will drive users (and developers) towards a more-standards-compliant browser with better performance.

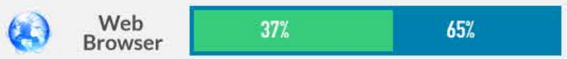
Outside the enterprise environment that will certainly be the case. Public applications which rely on the proprietary features of Internet Explorer are already few and far between, and the adoption of Edge will hasten their demise, but in the corporate environment developers may be slower to adjust.

Early releases of Edge are demonstrably faster than Internet Explorer, and enterprise users who have to switch back to IE for internal applications will not be slow in expressing their annoyance, so we expect to see an upturn in development targeting the latest web browsers over the next 12 months.

# USER UPGRADE INERTIA DOMINATES DESKTOP PLATFORM CHOICES

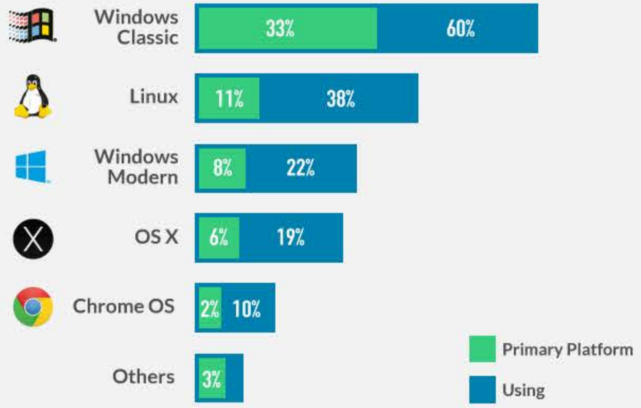
## 65% of desktop developers target the browser and 60% still target classic Windows – it's where the users are

37% of desktop developers prioritise the web



% of desktop developers using and prioritising the browser

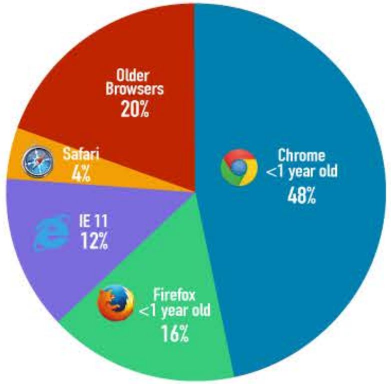
Only 22% of desktop developers build modern windows apps



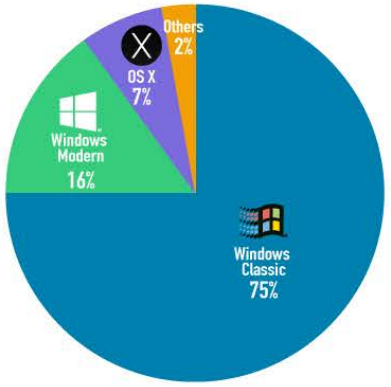
% of desktop developers using and prioritising each platform

**80%** of web page visits are from a very recent browser

% of page views, source: StatCounter



**75%** of desktop computers are running a version of Windows that was launched 6+ years ago



% of web users, source Net Applications

Source: Developer Economics - State of the Developer Nation Q3 2015 | vmob.me/DE3Q15 | Licensed under CC BY ND | Copyright VisionMobile

The success of Edge is intimately tied to the success of Windows version 10, so developers have even more reason to keep a close eye on the progress of Microsoft's next-generation platform.

## Windows still ruling the desktop

Three quarters of desktop computers are running Windows 7 or earlier, with Windows 8 claiming another 16% of the market leaving Apple's OSX with only 7% (the remaining 2% being occupied by various flavours of Linux and more-obscure platforms).

It's clear that the developer-friendly additions to Windows 8, such as the integrated store and cross-platform development system, have not proved as attractive as Microsoft had hoped. Attempting to ape the success of Apple's desktop store, and integrate mobile and desktop platforms unto a common user experience, has failed to create a significant developer ecosystem. In part that can be attributed to the failure of Windows 8 itself - developers will only create applications for a platform which is in use, but it also bodes badly for the Windows Universal Platform which further integrates development across device types.

Those numbers come from Net Applications, but go a long way to explaining the dominance of Windows as a developer target. In fact, the proportion of developers aiming at Windows and OSX reflects the market share quite accurately, so it's the anomalies across the minority platforms which are most interesting.

## The Linux desktop might never arrive, but the Linux desktop developer is already here

Linux, for example, claims less than 2% of the desktop market – lumped in with the "other" category – but 11% of desktop developers report that Linux is their primary target. Linux is also targeted by a disproportionate number of developers as an additional platform; one for which they are also creating applications, again despite the lack of Linux on the desktop, and this is worthy of some examination.

Linux distributions are generally open source, and often created as a community effort involving a large number of developers who volunteer their time for the good of the community. These "hobbyist" developers often work professionally in software development, but enjoy contributing to open projects in their spare time. The contribution of each developer is small but there are, by necessity, a lot of them. The same can be said of popular applications for Linux – such as the Gimp image editor, or LibreOffice, both of which are community projects involving a very large number of developers.

We examined this type of development activity in some detail last year, in our Developer Segmentation 2014 report[1] from where we can see these Linux enthusiasts as a classic example of Hobbyists according to our segmentation model.

In that report we also identified a significant segment of "Explorers" learning mobile development to maximise their future opportunities. There's an equivalent group targeting desktop Linux - developers creating applications for the desktop, but viewing the platform as a small-scale server for specific applications. The scalability of Linux invites this kind of experimentation, allowing developers to try out applications or even just hone their skills towards large-scale developments.

## Chrome – the OS that just might

Equally anomalous is the developer support for Chrome OS, particularly given that most Chrome OS applications will run equally well in a browser. Chrome OS is Google's thin client operating system, designed to delegate work to the cloud whenever possible and offering only limited functionality without connectivity.

[1] http://www.visionmobile.com/product/developer-segmentation-2014/

2% of the developers in our survey said they are specifically targeting Chrome OS, as opposed to creating applications for browsers which will also work on Chrome OS devices, which is surprising when the OS itself has negligible market share. Perhaps even more remarkable is the 10% of developers who include Chrome OS in their list of supported devices – showing a high level of interest in the platform even if that hasn't yet resulted in boosted sales.

> *2% of the developers in our survey said they are specifically targeting Chrome OS*

The developer interest in Google's thin client is in part driven by curiosity, as a new technology developers are keen to understand its capabilities and limitations, and Google has worked hard to make it easy for developers to package applications for its platform. Taking a web app and bundling it for Chrome OS is trivial, but the fact that developers are doing it still demonstrates a growing interest in the use of thin clients.

Enterprise users have much to gain from cloud-based desktops like Chrome OS. They may not be suited to some jobs, where local processing or off-line computing is essential, but there are numerous roles where connectivity can be assumed and the advantages of elastic computing outweigh the limitations, where cloud computing can usefully be deployed.

We expect to see interest in Chome OS growing slowly, alongside other cloud-base (thin client) computing systems, as developers learn more about the capabilities (and limitations) of such systems, and start to create and deploy applications.

# 5 EVERY CLOUD HAS A LINING, BUT THEY AREN'T ALL SILVER

## Public clouds might be the way of the future, but as far as developers are concerned they're not yet the way of the present.

Almost half of developers are hosting their apps in private clouds, well away from the public infrastructure which grabs all the headlines. Even Amazon Web Services, the biggest cloud by far, can only boast of being the primary platform for 16% of software developers, while 44% of them are happier to keep their clouds safely at home.

Not that developers are trying to avoid cloud computing – the advantages are well understood, but concerns linger about security and resilience. In many instances the convenience of self-hosting still outweighs the advantages of the public cloud. The availability of cloud environments is also a factor, enabling enterprises to realise many of the advantages of cloud computing within their own infrastructure, and ensure their applications will be cloud-friendly when the advantages of a hosted solution become irresistible.

Future proofing applications also means using a cloud-friendly development language, but it seems clouds are less multilingual than they might appear. All the popular cloud services support a wide range of languages, so we should expect to see a fairly-uniform spread of development choices, but in fact we can see distinct preferences as users of specific languages show a demonstrable preference for particular hosts.

In part this may be historical – companies who have shown a public backing for a language, or language style, may attract developers comfortable with that decision – but far more important is the support that cloud platforms provide. While languages may be common the APIs made available through those languages differ between providers, and features available on one cloud may appeal to users of a specific language.

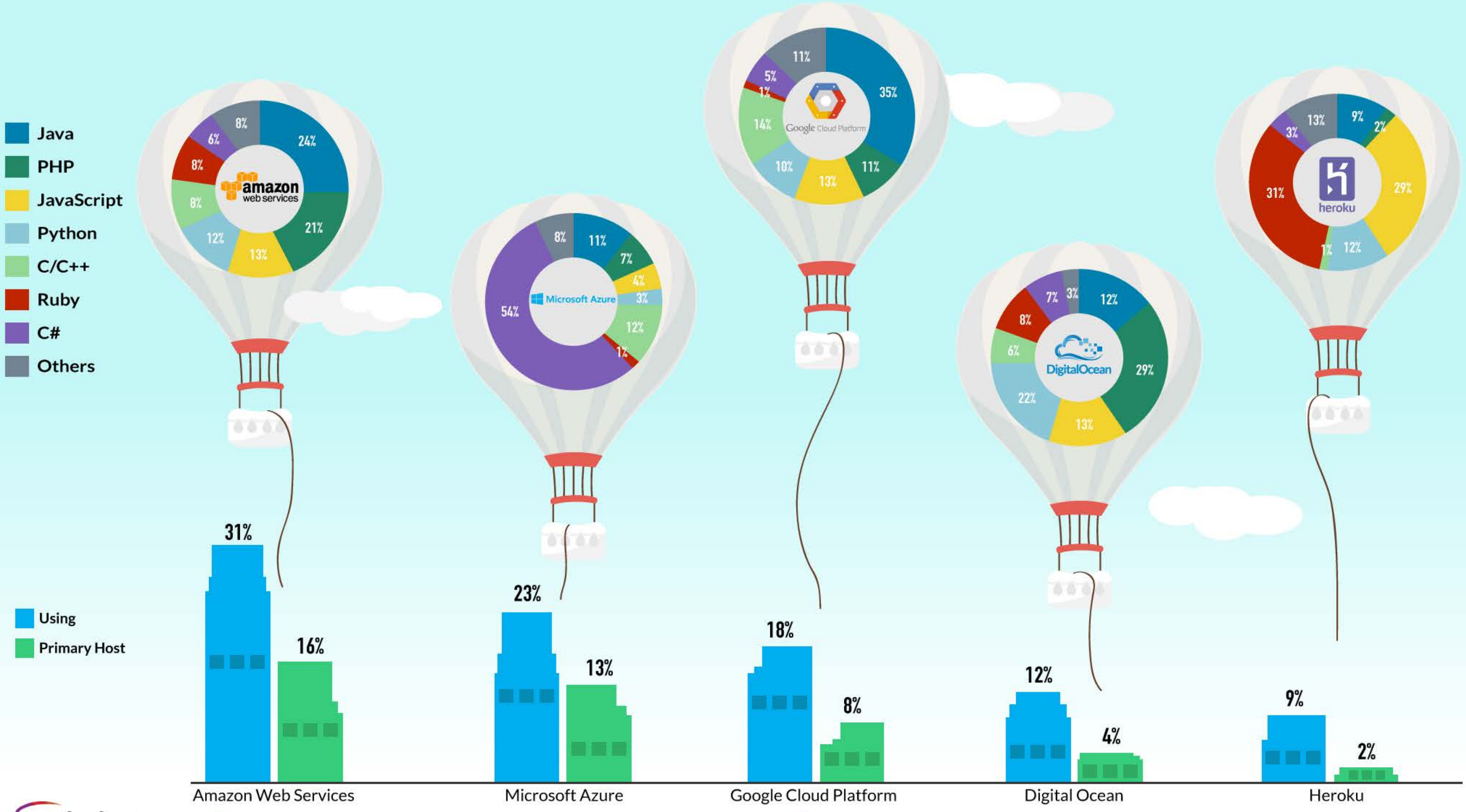
> *Even Amazon Web Services, the biggest cloud by far, can only boast of being the primary platform for 16% of software developers, while 44% of them are happier to keep their clouds safely at home.*

Google's PaaS offering, Google App Engine, only supports Java, Python, PHP & Go, and the Java VM supports the latest frameworks and Google offers extensive tutorials showing how to get the best out of the App Engine with Java. Heroku offers dynamically scalable services well suited to scripting languages, such as Ruby and JavaScript, so it isn't surprising that we see those kinds of language dominate that cloud.

Developers don't always get much say in the choice of cloud provider; previous relationships and pricing are significant drivers, but it's clear that they are pushing their employers to use cloud providers suited to their development language and style.

# AZURE VS. HEROKU VS. AWS: IT'S ALL ABOUT THE LANGUAGE

## 54% of Azure developers prefer C# while 60% of Heroku developers prefer Ruby or JavaScript, AWS is the place for all



vision mobile

% of developers using and prioritising each hosting platform

% of developers primarily using each language by primary hosting platform

Source: Developer Economics - State of the Developer Nation Q3 2015 | vmob.me/DE3Q15 | Licensed under CC BY ND | Copyright VisionMobile

## Cloud computing starts at home

Amazon is by far the largest provider of cloud services, building on their own infrastructure the former book store now represents well over half the cloud industry, but despite that dominance (which covers SaaS, PaaS, and IaaS) we see only 16% of developers using Amazon as their primary cloud hosting platform. Almost a third are using Amazon in some capacity, but in terms of developer preference Amazon is only a shade more popular than Microsoft Azure.

This anomaly can be explained by looking at Amazon's biggest customers for cloud services. The hugely-popular Netflix service, which provides streaming video to 50 million customers worldwide, is hosted on Amazon's cloud. Amazon Instant Video, a competitor to Netflix, integrates with Amazon's Cloud Drive to let users store (and stream) their own videos, again bolstering the numbers, while popular internet services like Pintrest and Yelp further contribute to Amazon's dominance.

Amazon may dominate the public-cloud industry, but when it comes to cloud development the most-popular hosting option is to keep things in-house.

> *In terms of developer preference Amazon is only a shade more popular than Microsoft Azure.*

Well over half of cloud developers are using a private cloud, hosting their own cloud to realise many of the benefits without relinquishing control, despite the plethora of public options available. Almost half (44%) say that their primary development platform is their own cloud, as enterprises take cautious steps towards diving into public clouds.

To most developers the key advantage of cloud computing is scalability, or (more accurately) elasticity. Applications with varying demand are well placed to take advantage of cloud computing, and the flexible infrastructure yields immediate (and obvious) benefits, but it is less obvious how more consistent enterprise applications benefit from cloud computing.

Public clouds are inherently more efficient. The advantages of centralised power management, physical security, and redundancy in infrastructure such as processing, storage and networking, enable cloud providers to promise unparalleled uptime, but those advantages are still ethereal to companies more familiar with buying boxes and running server farms.

> *Amazon may dominate the public-cloud industry, but when it comes to cloud development the most-popular hosting option is to keep things in-house.*

Integrating with legacy systems is also easier when the hardware is in-house. Local databases or control systems may have proprietary interfaces which can only be accessed over the local area network, or via a proxy system which might be complex to develop and provide an additional point of failure.

Over time we expect to see more developers targeting public cloud platforms, as concerns over security fade, legacy systems get updated, and the resilience of cloud computing has more opportunity to prove itself. Companies hosting cloud applications locally should be able to move onto a public cloud without great upheaval, depending on the development environment and the eventual host selected.

## Amazon clouds speak everyone's language

It's no great surprise that Amazon's Web Services provides the greatest linguistic variability, as the largest provider of cloud services it is selected by companies who perhaps worry more about reputation than language support. Amazon has always portrayed itself as language (and platform) agnostic, and it certainly provides a breadth of support without seeming to back any specific platform or business model.

Java is the most popular language, followed closely by PHP, but even those two aren't obviously predominant. Every other cloud platform can be said to have one or two dominant languages, and a crowd of less-popular alternatives with very-low usage figures, but Amazon customers clearly aren't selecting the platform on the basis of language support and thus are developing across the board.

The Amazon usage figures are, therefore, a useful reflection of the languages used by cloud developers, rather than those favoured by Amazon developers.

## Microsoft clouds attract Microsoft developers

In stark contrast to Amazon we can see that well over half the developers targeting Microsoft Azure are coding in C#. C# was developed by Microsoft around the turn of the century, and has the avowed support of the company. Envisioned as a cross-platform language scalable enough for embedded and server development, C# is popular for cloud development, though few would argue it represents half the industry as the figures for Microsoft Azure would seem to indicate.

Microsoft has always been very good at attracting developers, to desktop and server platforms at least. In part this can be attributed to the history of the company, which was built on the premise of facilitating open innovation (though MS-DOS), but Microsoft has also fostered that reputation with investment in development tools and support systems to keep developers happy. C# offers the most modern of development environments, and by linking that in with the Azure cloud Microsoft has (again) levered its market position in one field into an enviable market presence in another.

Azure customers are under no obligation to use C#, and as a predominantly-Windows platform the cloud service supports a very wide range of development options, so the fact that C# developers make up more than half the Azure customer base demonstrates how successful the Microsoft strategy has been.

## Heroku – exploiting the niche carved by developers in a hurry

Heroku is another example of a language specialist lending support to a cloud service. In 2011 the chief designer of the Ruby programming language joined Heroku as the company's Chief Architect. That brought a huge amount of Ruby knowledge to the highest levels of the company but it also brings the philosophy behind Ruby to the architecture of Heroku, thus attracting a large number of Ruby developers.

Ruby is now the most popular language on Heroku, though JavaScript is a close second. Between the two they account for almost two thirds of developers hosted on Heroku, which is also the only platform without a significant number of C/C++ developers on board.

Servicing the needs of those working in script, Heroku has developed APIs to enable dynamic elasticity from high-level calls. That would be cumbersome in a low-level language such as C, but enables scripted languages to operate with great flexibility.

Heroku provides a clear example of how a cloud host can add value by morphing the service into something suitable for a specific group of developers. Not only creating a platform that is attractive to those working in specific languages, but also discouraging them from moving with platform-specific features they can't get elsewhere.

# 6 SMART HOMES ARE STILL HOME TO THE SMART MONEY

The Internet of Things is moving rapidly from a theoretical concept to a material reality. Since our last State of the Developer Nation report we've seen two new IoT radio networks built out over London (Sigfox and Weightless) and new radio standards published (LoRa and Cellular IoT).

Our IoT Landscape report documents many of the latest developments, including real products and businesses emerging to realise the value of connecting up Things[2].

Interest in IoT is still focused on the smart home, but retail, industrial, and wearables are also attracting attention from developers keen to turn interesting ideas into profitable products. At the other end we find the Smart City, Connected Car and Medical fields aren't getting the same attention, despite high-profile deployments in those areas.

Hand-in-hand with that we can see the majority of developers are targeting consumers: the obvious market for smart home and wearable technology, but more than a quarter are still not sure who their eventual customer will be. Those uncertain developers tre the hobbyist community, which is experimenting with IoT, and companies researching how they can apply their existing skills and experience to the IoT sphere.

There's also a trend away from hardware development, though it's more obvious in some vertical industries. Developing hardware is expensive, and hard, as many crowdfunded projects have discovered. The risks involved in creating new hardware are much greater than software, requiring more investment and a slower return, so it's no surprise that, wherever possible, developers are creating systems to utilise existing hardware and firmware, rather than creating their own.

> *Smart homes are still the primary focus for developers, buoyed up by the prospect of new devices supporting Apple's HomeKit, and Google's Brillo.*

## Smarter developers are targeting the (even) smarter home

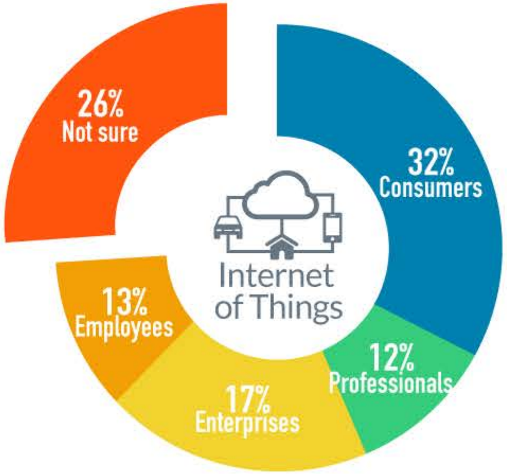
Smart homes are still the primary focus for developers, buoyed up by the prospect of new devices supporting Apple's HomeKit, and Google's Brillo. The two companies have drawn a great deal of attention on home automation, both from the media and developer communities, and this is reflected in the fact that Smart Home products top the list of targeted verticals.

[2] http://www.visionmobile.com/product/iot-developer-landscape-2015/

# IOT: LOTS OF SOLUTIONS STILL LOOKING FOR THE RIGHT PROBLEMS

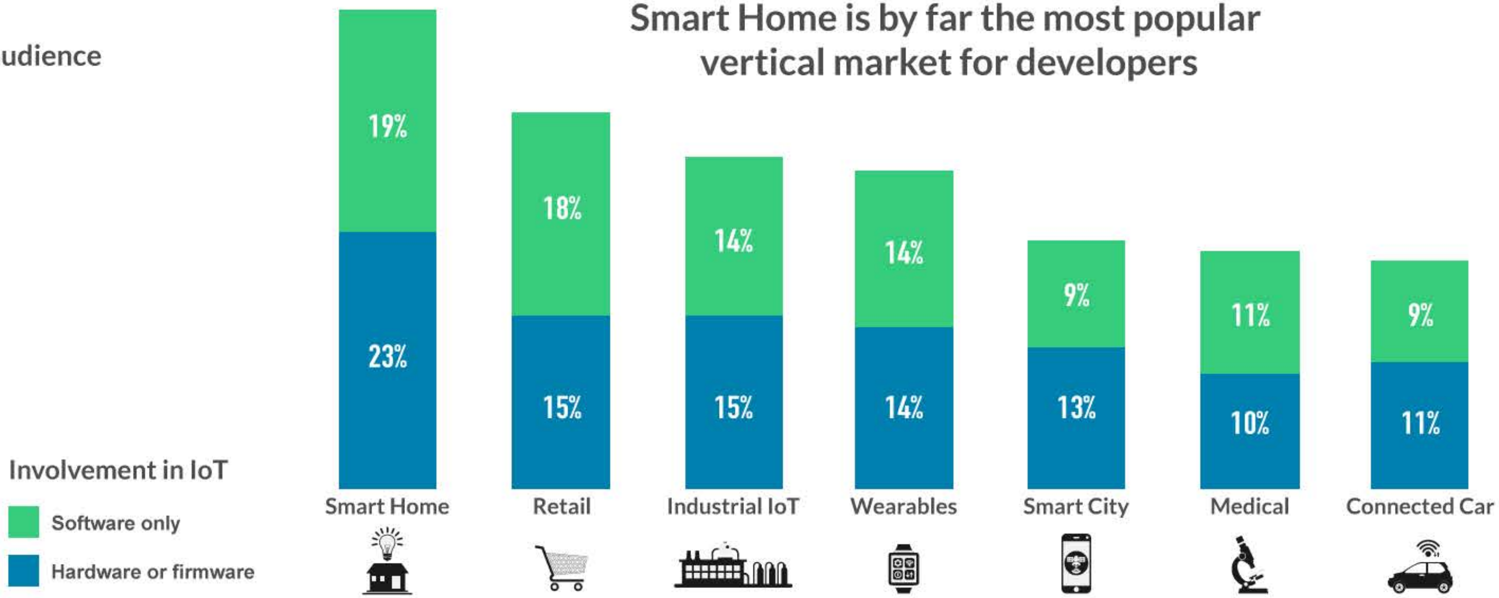
## Almost 50% of IoT developers now build software-only solutions, the rest are building "Things" but who are they building them for?

### 26% of IoT developers aren't sure of their audience



% of IoT developers targeting each audience

### Smart Home is by far the most popular vertical market for developers



% of IoT developers currently targeting at least 1 vertical market

Source: Developer Economics - State of the Developer Nation Q3 2015 | vmob.me/DE3Q15 | Licensed under CC BY ND | Copyright VisionMobile

The lack of interest in Smart City is more surprising, as governments and municipalities have gone a long way to promote IoT with developers. In London, the government's Digital Catapult initiative has gone as far as to help fund a city-wide IoT network (using the Weightless standard), providing connectivity for Smart City projects across London, and the Catapult continues to fund and encourage projects.

But despite the funding and political endorsement, Smart City projects are still slow to develop and deploy, often requiring developers to work with local bureaucracies which may result in a clash of cultures. Google and Apple are going to great lengths to promote Smart Home development, which provides a lower barrier to entry than developing products for municipal deployment.

Those two companies have also done a lot for wearables, though the promotion of the Apple Watch and Android Wear platforms. Not only do these devices provide a consumer base into which applications can be sold, but the integrated development environments and community support are attractive to developers who may otherwise be reluctant to explore a demonstrably-unproven market.

Medical and Connected Car are proving less popular, certainly because both suffer from significant barriers to entry which haven't always been obvious. Medical authorities are starting to pay attention to uncertified applications providing medical advice, while car manufacturers have proven too conservative for many observers who had hoped we'd all be using integrated displays and connectivity by now. Even if Apple's CarPlay and Google's Android Auto are wildly successful, they'll take years to penetrate the global fleet of cars (the average car in the US is 11.4 years old). These verticals will develop, and will prove hugely important in the years to come, but for the moment they aren't attracting the developer community.

## We know what we're doing, but not for whom we're doing it

Consumers are clearly still the target for IoT, with almost a third of developers saying that they're chasing the consumer market, but that still leave a large proportion (more than a quarter) unsure as to whom their eventual audience will be.

Alongside that uncertainty is a significant amount of internal development, creating applications and services for "employees in my company". That trend reflects an increased understanding of IoT, and how it can create value within existing organisations.  One could suggest that the first wave was driven by engineers sitting at home, and looking around to see what intelligent Things could do, but now many of those engineers are at work and seeing more things that can be improved with greater intelligence.

There is also an argument that workplace IoT is easier to implement as practices can be changed by dictate, while consumer products are always expected to work around existing arrangements.

These developer trends will certainly be followed by increased industry activity in the next 12 months, as companies deploy the internal products they are now developing, and their competitors see the value they're gaining.

Even more interesting, perhaps, is the proportion of those claiming to be "not sure" of their eventual market, but developing systems anyway. A good deal of this will be research and experimentation, as the value of many IoT systems has yet to be properly understood, and companies are developing concepts which may prove applicable to multiple markets. The IoT industry is still developing, and this uncertainty about eventual audience is a sign that developers, and the companies they work for, are starting to understand that.

## Hardware is hard, and best avoided

Developing Things is a challenge, and one that shouldn't be undertaken lightly. Creating mass-produced hardware is expensive, involving deals with suppliers (often in the Far East) and running up production lines which can't easily be altered or suspended. 3D printing and low-cost electronics has made the production of prototypes deceptively easy, but the transition into mass production remains as complicated as ever.

Despite the challenges we can see that a significant number of IoT developers are working on hardware, and the firmware it runs. Those writing firmware might be part of product development – all hardware needs firmware – but they could also be involved in product enhancements enabling new functionality, or moving hardware from one vertical industry to another.

> *More than half of the professional mobile developers sampled in our survey are also involved in IoT*

Most encouraging is the growth in those not creating hardware or firmware ("neither"), as it reflects IoT developers creating applications for existing Things. The trend is particularly obvious in retail, where more than half of developers are creating applications rather than infrastructure, reflecting the mature nature of embedded devices used in retail – tills, bar-code readers, smart tags, and beacons are already being used in a retail environment, so developers can focus on new ways to utilise those Things rather than having to create new ones.

The same trend can be seen in medical, though this can also be partly attributed to the growth in wearable technology which is often identified as being of medicinal value. Wearables also show a less pronounced leaning towards application development, thanks to the coalescing of smartwatch standards around the market leaders Apple and Google.

That's a trend which we expect to spread into the other vertical markets, as hardware (and firmware) become polarised with minor players being forced to adopt common platforms. That will let developers focus on applications rather than infrastructure, leading to a steady decline in the proportion working on hardware development.

## Mobile Things are attracting mobile developers

More than half of the professional mobile developers sampled in our survey are also involved in IoT, demonstrating the cohesion between two fields. Just over a quarter of them are professionally involved in IoT while another 35% are taking an interest in the field as hobbyists. This overlap is partly attributable to the use of mobile handsets as an interface to the Internet of Things; many IoT projects require significant input from mobile developers, but it also reflects the commonality in tool-chain, development processes, and resource limitation, which make skills in mobile readily applicable to IoT projects.

> *The trend is particularly obvious in retail, where more than half of developers are creating applications rather than infrastructure.*

Hobbyist mobile developers are even-more drawn to IoT, though mostly as an extension to their hobby. The wide availability of Arduino hardware has lowered the hobbyist barrier to entry, and with hardware costing only a few dollars, and tutorials freely available, there is little reason not to expect this proportion to grow steadily.

# 7 DEVELOPER REVENUES: THERE'S MORE MONEY IN THE CLOUD

Having expanded our survey across mobile, desktop, IoT and cloud developers, we can compare revenues for those working in different areas.

We need to be careful when making these comparisons because, although sales of standalone mobile and desktop apps can be easily separated and compared, it's not always meaningful to split a cloud service from its client apps in terms of revenue generated. Similarly an IoT device generally needs some kind of cloud service to provide the "Internet" to go with the "Thing". However, by also looking at the most popular and most lucrative revenue models for each area of development, we can still make some useful comparisons.

## Cloud developers do better

The high-level view is that IoT market is not yet large enough to sustain many serious businesses, mobile apps are still not making any real money for more than half of those involved and desktop apps are only slightly better. Cloud services developers have the best odds of making money although 43% of those interested in revenues still fall below our "app poverty line" of $500 / month. This outperformance of cloud services developers is interesting because there must be some kind of client, be it mobile app, desktop app or IoT device. Since IoT developers are typically not making much money we can conclude that the mere presence of a cloud service connected to a mobile or desktop app is improving the odds of success.

## Mobile developers are still going for the low-hanging fruit

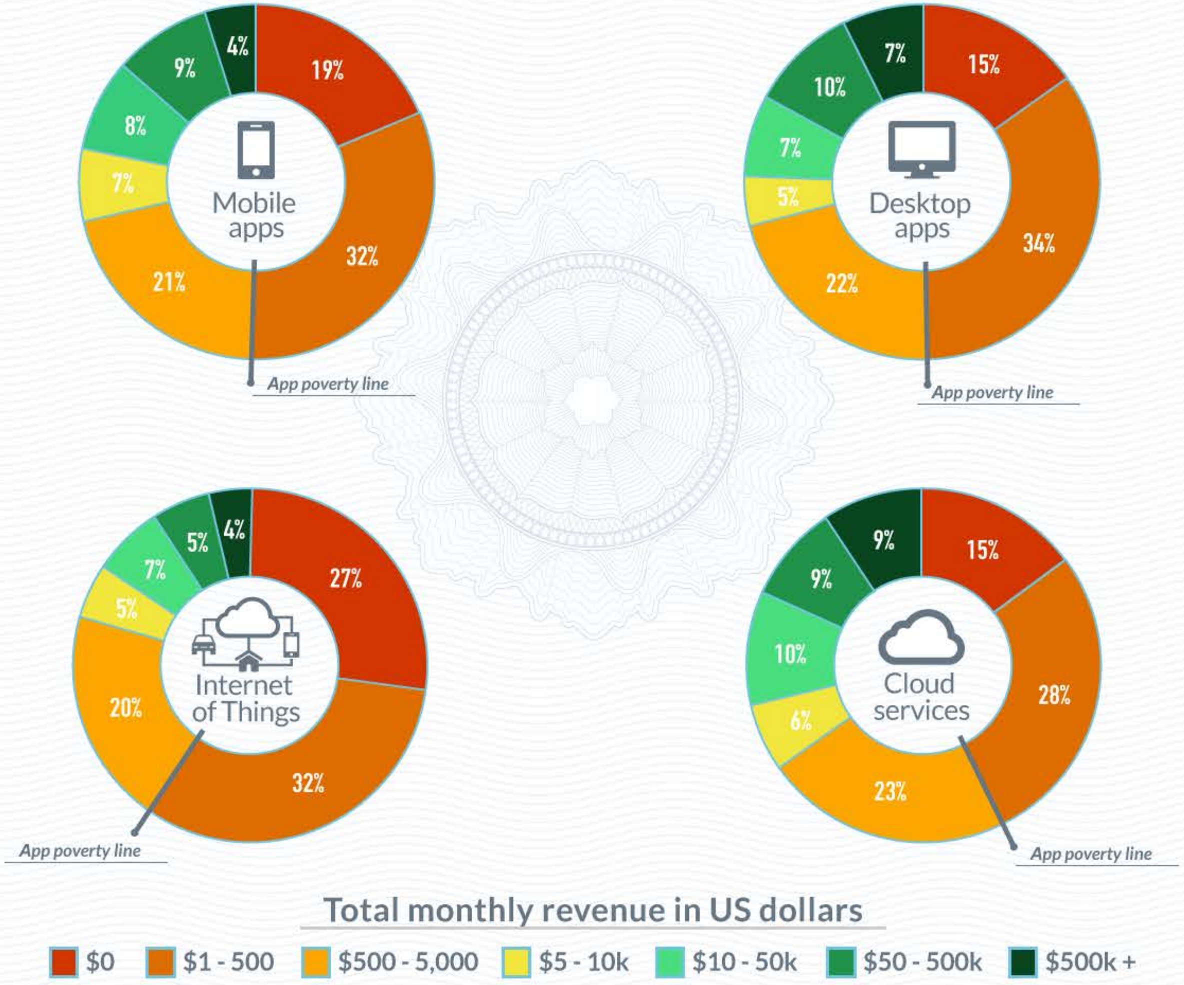
51% of mobile developers are still below the app poverty line. A major cause of this is that most of them still persist in trying to make money via the simplest revenue models to implement - paid downloads and advertising. We have repeatedly reported on the way paid downloads are not working for developers. The vast majority of revenue through the app stores has been via in-app purchases for some time now and the percentage keeps increasing. There's still the occasional paid app hit which seems to keep this dream alive in developer minds but using this model is no more than a high risk gamble in most cases. Advertising on the other hand can be consistently successful. The issue is that it's only worthwhile for apps with gigantic audiences that remain engaged for a long time. To make enough money to support an app business with mobile ad networks requires millions of users that engage with the app frequently. Apps with that level of user engagement can probably do better with direct advertising deals than an ad network anyway. A developer interested in making a little extra money with a side project can integrate an ad network easily but it's not likely to generate much revenue.

# DEVELOPER REVENUES: CLOUD SERVICES ARE THE MOST LUCRATIVE, INTERNET OF THINGS THE LEAST

## 28% of Cloud Services developers make more than $10k per month vs. 15% for Internet of Things developers

### Total monthly revenues by development area

% of developers interested in revenues



### MOBILE APPS

Revenue Models

**Most popular:**

Advertising
Paid downloads

**Most lucrative:**

e-Commerce (non-digital goods)
Affiliate or CPT programs

### DESKTOP APPS

Revenue Models

**Most popular:**

Advertising
Contract work

**Most lucrative:**

Royalties or licensing
Selling services to developers

### INTERNET OF THINGS

Revenue Models

**Most popular:**

Selling physical products
Software licensing

**Most lucrative:**

Software licensing
Selling physical products

### CLOUD SERVICES

Revenue Models

**Most popular:**

Contract work
Selling physical products

**Most lucrative:**

Royalties
Licensing client software

Source: Developer Economics - State of the Developer Nation Q3 2015 | vmob.me/DE3Q15 |

By contrast, the developers that are making the most revenue through mobile apps are using them as a channel to sell real-world goods and services. As we'll see in Chapter 8, e-Commerce is where future mobile fortunes will be made. Consumer purchasing behaviour is shifting to mobile and the platforms are making it a much smoother process. Of course, having (or working for) a real-world goods or services business may not be what most developers want. The next most lucrative revenue models are affiliate or Cost Per Install (CPI) programs. Affiliate programs are essentially a way of advertising for the real-world goods and services companies - the developer gets paid a commision when a purchase is made. However, they can be integrated in lots of different kinds of app. For example, a travel guide app, rather than charging for a content download, could integrate an affiliate program from someone that sells flights or hotels. CPI programs are just advertising for other apps where the developer gets paid only when apps are installed. These pay a lot better than simply showing ads for other apps on a cost per impression (CPM) or cost per click (CPC) basis, although this does depend on being able to reach the right audience for the target apps.

> *59% of IoT developers are earning less than $500 per month and 79% less than $5,000 per month.*

## Desktop developers can succeed by helping other developers

Overall, desktop developers don't do much better than mobile developers, with 49% falling below the app poverty line. This is unsurprising considering that their 3 most popular revenue models are the same as for mobile developers, except that contract work comes 2nd, with paid downloads 3rd, while the order is reversed on mobile. Working for hire is much lower risk than building your own products but not a route to riches.

Desktop developers are having the most success with royalties and licensing. A revenue model that usually requires mature and widely-known intellectual property to be very successful. It's not a very easy model to get started with but fits the more established market well. More interesting is that the 2nd most lucrative model for desktop developers is selling services to their peers. Developers represent a very large group of desktop power users who need a lot of software to run their businesses. The current trend is towards Software as a Service offerings for everything from analytics and test automation to project management and version control. Almost all developers need these tools and most developers are experienced users with ideas for how their tools could work better.

## Selling 'Things' isn't working for most

The most popular revenue model for IoT developers is selling the physical products that run and work with their software and services. For smaller independent developers it has been very popular to crowdfund these projects in the past few years but most have found the realities of relatively small scale hardware production very tough. Software licensing is the 2nd most popular model. As mentioned with desktop software above, this is a difficult model to get started with. The market is not yet mature and too many people are building generic solutions to device connectivity and management problems. There are hundreds of developers trying to build platforms as pieces of technology for sale.

Even so, the most lucrative revenue models almost mirror the most popular ones at this stage of the market. This should not be taken as a sign developers have found the right models, it's just that nothing else is working yet either. 59% of IoT developers are earning less than $500 per month and 79% less than $5,000 per month. Those selling 'Things' will typically have fairly low margins on low volume production runs and those trying to build platforms will need a decent sized team of developers. Less than 10% of IoT developers are currently making enough revenue to support such a team.

# 8 PROFITABLE DEVELOPERS ARE SELLING STUFF, NOT EYEBALLS

Mobile applications have generally been viewed as an end in themselves, providing functionality or entertainment in exchange for a purchase price or embedded advertising, but the revenue generated by such processes is dwarfed by the quantity of e-commerce transactions being conducted through mobile applications.

At VisionMobile we consider e-commerce to be anything involving the sale of real world services or products, through an application accessed through a mobile device. Subscriptions (such as Netflix and Spotify) are a separate category, as are in-app purchases (of the kind beloved by mobile-games companies). Here we are interested in developers who are generating revenue by selling tangible services; a group which boasts the biggest proportion of profitable businesses, and those developers reliant on advertising; if which only a small proportion are bringing in sustainable revenue.

> *Revenue from advertising can't hope to match the numbers made by selling real things*

Advertising is still a significant source of revenue for some mobile applications: 17% of developers who rely on advertising are making a reasonable living, but that compares to 37% of those actually selling stuff to keep the money flowing, making it obvious that the future lies in transactions actioned from within an application.

Applications that sell physical objects or services, such as fast food (Just Eat) or hire cars (Budget, Hertz, etc) obviously bring in much greater revenue than those selling character upgrades or in-game artefacts. Revenue from advertising can't hope to match the numbers made by selling real things, even if the margins on physical and digital goods are comparable for scalable businesses.

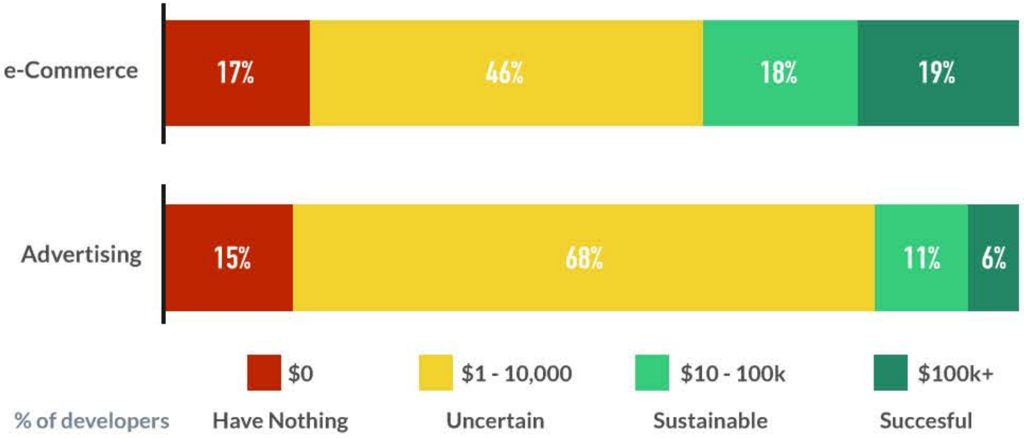
The e-commerce market is still dominated by innovative start-ups who have carved new business models by intermediating between consumers and producers – the Uber taxi service being the best known, though Just Eat is probably a better example as it matches buyers with existing businesses rather than independent contractors, but that's changing as more businesses take advantage of mobile e-commerce and established brands come to compete with the new intermediaries.

> *17% of developers who rely on advertising are making a reasonable living, but that compares to 37% of those actually selling stuff to keep the money flowing.*

# E-COMMERCE IS THE NEXT BIG BATTLEGROUND IN MOBILE

## Mobile commerce developers are more than 3x as likely to make $100k+ per month than those monetising with ads

% of developers interested in revenues, using e-Commerce or advertising revenue models



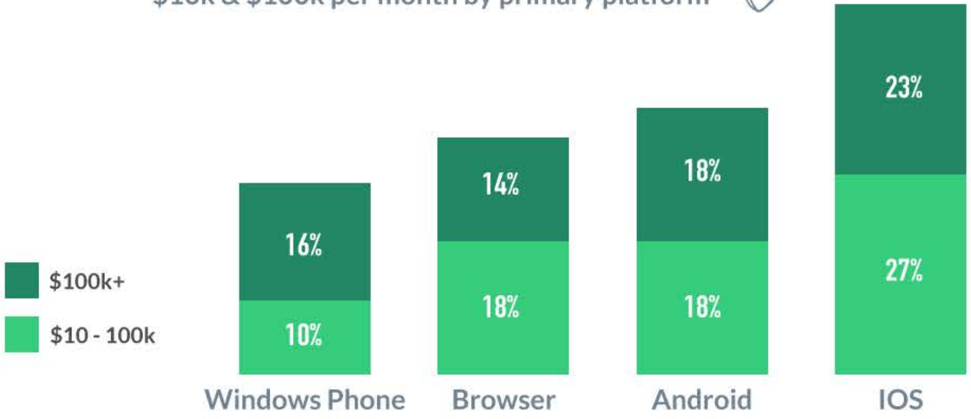
10% of mobile developers use e-Commerce as a revenue model

46% of mobile developers use advertising, the most popular revenue model

Mobile commerce revenues from native platforms are already overtaking those from the mobile web.

Internet giants everywhere want to be involved in the transactions.

% of e-Commerce developers making more than $10k & $100k per month by primary platform



vision mobile

Source: Developer Economics - State of the Developer Nation Q3 2015 | vmob.me/DE3Q15 | Licensed under CC BY ND | Copyright VisionMobile

Despite the obvious advantage of tangible commerce only around 10% of developers are trying to make money from it, with a much larger proportion (46%) looking to advertising for their revenue. Most of the rest are now making money from in-app purchases, selling additional levels or features within an application, making use of the payment-processing systems made available by the application stores.

Those processing systems are rarely used for physical payments, and it would be against the terms of use in most cases, so the owners (Apple, Google and Amazon) all have alternative mechanisms designed for such payments.

Amazon is something of an exception in this context, as the company sells digital content and subscriptions as well as physical goods (e-commerce), not to mention processing in-app purchases for developers using the Amazon app store. On the Android platform, Amazon provides a selection of native applications, most focused on selling (and rendering) a specific kind of digital content (mobile apps, music, videos, electronic books) and one connected to the main Amazon service through which the user can buy physical products sold by Amazon (e-commerce services), but all the Amazon apps are linked to the same payment mechanism ("1-Click").

That contrasts with Apple and Google who separate out their e-commerce payment systems as Apple Pay and Android Pay (née Google Wallet) respectively, requiring developers to integrate with those systems or take payments directly from the customers.

## Advertising is still bringing home the bacon for many

Almost half of mobile developers are still reliant on advertising for revenue, despite the fact that it's proving profitable for only a small minority. The vast majority; 83% of those who responded to our survey, say that advertising makes less than $10,000 a month – an amount we consider unlikely to sustain a professional business.

For many of them that won't be a problem – there's still a huge community of mobile developers who are working for beer money. These hobbyists are creating basic games, or niche applications tailored to their other interests, and slapping on an AdMob-sourced banner ad to cover the cost of their app-store registration and (perhaps) make enough money for drink or two. But here we are counting developers who've told us that they are interested in making money, even if they aren't generating much revenue yet they have aspirations to professional status.

A small minority (17%) are making decent money from advertising, in many cases doing direct deals with advertisers rather than going through the popular platforms such as AdMob and 4INFO. Direct advertising can embed content far deeper into the application, integrating it into the game or application being used. The advertising platforms have gone some way to enabling this kind of "native advertising", but it is clearly more effective when contextually integrated into the experience

This kind of native advertising is going to increase steadily, as advertisers find more ways to integrate their message within the mobile applications, but that will only be to the benefit of established developers and applications. Creating this kind of customised advertising is more time-consuming than a standard banner, requiring a commitment from both sides which will only be worthwhile if the application is going to achieve a decent level of success. For that reason the number of developers making money from advertising is unlikely to change much over the next year, though there will be a shift away from smaller and newer brands towards established players.

## iPhone users still spend more than anyone else

Half of the developers targeting iOS, who've chosen to make money from e-commerce, are pulling in more than $10,000 a month, reflecting the greater disposable income amongst iOS users, or perhaps their willingness to make payments through their mobile device. The dominance of the iPad in tablet markets is also significant, as e-commerce is certainly more prevalent on larger devices.

Amazon has previously reported that consumers bought more through tablets than phones, claiming that phones were often used for browsing content during the day, with purchases made on larger devices in the evening. Buyers seem more comfortable with a larger screen when entering credit card numbers, or checking delivery details, and are more willing to commit to purchasing when they can see all the details on one screen.

> *32% of developers targeting the mobile browser are making a reasonable income, with 14% topping $100,000 a month.*

That certainly applies to buying many physical objects, but services such as Uber take advantage of the mobility of the phone handset, and Just Eat bills itself as an impulse purchase – something which can be done instantly and easily, despite the difficulty sometimes experienced in navigating a fast-food menu on a diminutive phone screen.

Increasing screen size, and resolution, is reducing the disparity between the phone and tablet experience, and users are getting more comfortable with buying goods and services from their phone, so any advantage iOS has gained from dominance of the tablet market will likely be short lived.

The fact remains that iPhone users are, on average, richer than those with an Android device, and are thus going to spend more on goods and services of all kinds. The quantity of Android devices will mitigate against this - Android users may spend less, but as their numbers increase we expect to see the revenue difference between the platforms shrink. Apple will remain the most-profitable platform for a while yet, but as e-commerce spreads from luxury to commodity goods the quantity of Android users will enable it to challenge that dominance.

## Insert coin to continue

Developers targeting iOS might be doing best out of commerce, but they are far from alone. Well over a third of Android developers looking to sell tangibles are making a sustainable income (considered to be more than $10,000 a month), and a quarter of those creating apps for Windows Phone are doing equally well, in fact the most surprising thing is that those targeting the mobile web aren't doing better.

32% of developers targeting the mobile browser are making a reasonable income, with 14% topping $100,000 a month, which is better than Windows Phone but not as good as Android. Selling things through web sites is old news, so it might seem that the business models should be well-defined and those who can't make a living will have already moved on.

> *Half of the developers targeting iOS who've chosen to make money from e-commerce are pulling in more than $10,000 a month, reflecting the greater disposable income amongst iOS users.*

The reality is that users are still reluctant to buy goods using a mobile phone, and the sometimes-substandard experience offered by the mobile web goes a long way to undermine what confidence the users have. They may browse using their phone, but when it comes to buying users will often turn to a desktop platform connected to a keyboard and large screen.

The numbers only show those developers who are 'targeting' the mobile browser, as opposed to web developers who happen to have coded an e-commerce site well enough to work effectively on a mobile browser. It seems likely that many of those developers are creating commerce sites designed to take advantage of the impulse purchase possible on a portable device, but have yet to gain traction.

The difference between a native application and a well-designed web site are narrowing, and mobile devices are getting much better at rendering complex content. The traditional barrier to mobile e-commerce; the entering of credit card details, is being addressed by both Chrome and Safari (both of which will autocomplete credit card numbers, the former by default), and as the platform owners integrate their payment systems with their browsers the process should be further simplified. This won’t have an immediate impact on the figures for mobile browsing but will accelerate the improving trend.

# METHODOLOGY

Developer Economics 9th edition reached an impressive 13,000+ respondents from 149 countries around the world. As such, it is the most global research on mobile, desktop, IoT and cloud developers combined ever conducted. This report is based on a large-scale online developer survey designed, produced and carried out by VisionMobile over a period of five weeks between May and June 2015.

Respondents to the online survey came from over 149 countries, including major app and IoT development hotspots such as the US, China, India, Israel, UK and Russia and stretching all the way to Kenya, Brazil and Jordan. The geographic reach of this survey is truly reflective of the global scale of the developer economy. The online survey was translated in 7 languages (Chinese, French, Portuguese, Japanese, Korean, Russian, Spanish) and promoted by more than 70 leading community and media partners within the app development industry.

To eliminate the effect of regional sampling biases, we weighted the regional distribution across 8 regions by a factor that was determined by the regional distribution and growth trends identified in our App Economy research. Each of the separate branches: mobile, desktop, IoT and cloud were weighted independently and then combined.

The survey gathered responses from developers across mobile platforms including Android, Amazon Fire OS, BlackBerry 10, Firefox OS, iOS, Java ME, Jolla Sailfish, Mobile Browser, Tizen, Windows Phone, Windows 8 and Ubuntu Phone.

To minimise the sampling bias for platform distribution across our outreach channels, we weighted the responses to derive a representative platform distribution. We compared the distribution across a number of different developer outreach channels and identified statistically significant channels that exhibited the lowest variability from the platform medians across our whole sample base. From these channels we excluded the channels of our research partners to eliminate sampling bias due to respondents recruited via these channels. We derived a representative platform distribution based on independent, statistically significant channels to derive a weighted platform distribution. Again, this was performed separately for each of mobile, IoT, desktop and cloud, using targeted vertical markets rather than platforms for IoT and cloud hosting providers for cloud.

As we have shown in our Developer Segmentation[3] report, there is no average developer: Our outcome - based segmentation model of eight developer segments shows that the choices and views of developers may vary wildly according to their desired outcomes from their development activity. Hobbyists, who just want to have fun, and Explorers, who are learning and testing the market, think very differently as compared to professional developers such as Hunters, who are after direct app revenues, and Product Extenders, who are using apps and digital services to promote their other products. We have therefore also weighted our results to minimize sampling bias for segment distribution across our outreach channels to derive a representative developer segment distribution. By combining the regional, platform and developer segment weighting we were able to minimise sampling biases due to these factors. All results in the report are weighted by main platform (or market), region and developer segment.

[3] http://www.visionmobile.com/product/developer-segmentation-2014/

distilling market noise into market sense