The Earth is Flat Still

Cartography on the Web
Adam Nemeth - @aadaam

Today

- Introduction to Web Cartography
- Do-Your-Own Map Engine - Why you should, why you shouldn't, why I did and how to do it
- What can we learn by building our own map engine?

Why I am here today?

- In 2010-2011 I did a project
- I learned some things there
- That may be useful to you as well

What I won't talk about

- My newest startup, or anything close
- jQuery, CoffeeScript or any technology
- bugs

Part I: Cartography

Why do we need maps?

- To find our place
- Wasn't an issue in the Middle Ages
- Medieval people didn't believe the Earth was flat - they just didn't care!
- Then we learned how to build ships which don't sink

Problem: the Earth is not Flat

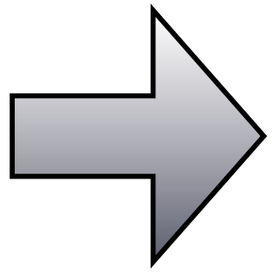
First Solution: Globe

Practical problems

- A globe effective in use to navigate between Győr and Budapest on Motorway - ⌀ 9 m
- A globe usable to navigate in Budapest - ⌀ 299.57 m
- Length of *Titanic*: 269 m

How to flatten the Earth?

Mercator Projection (mug projection)



Mercator Projection (mug projection)

Problem: distortion

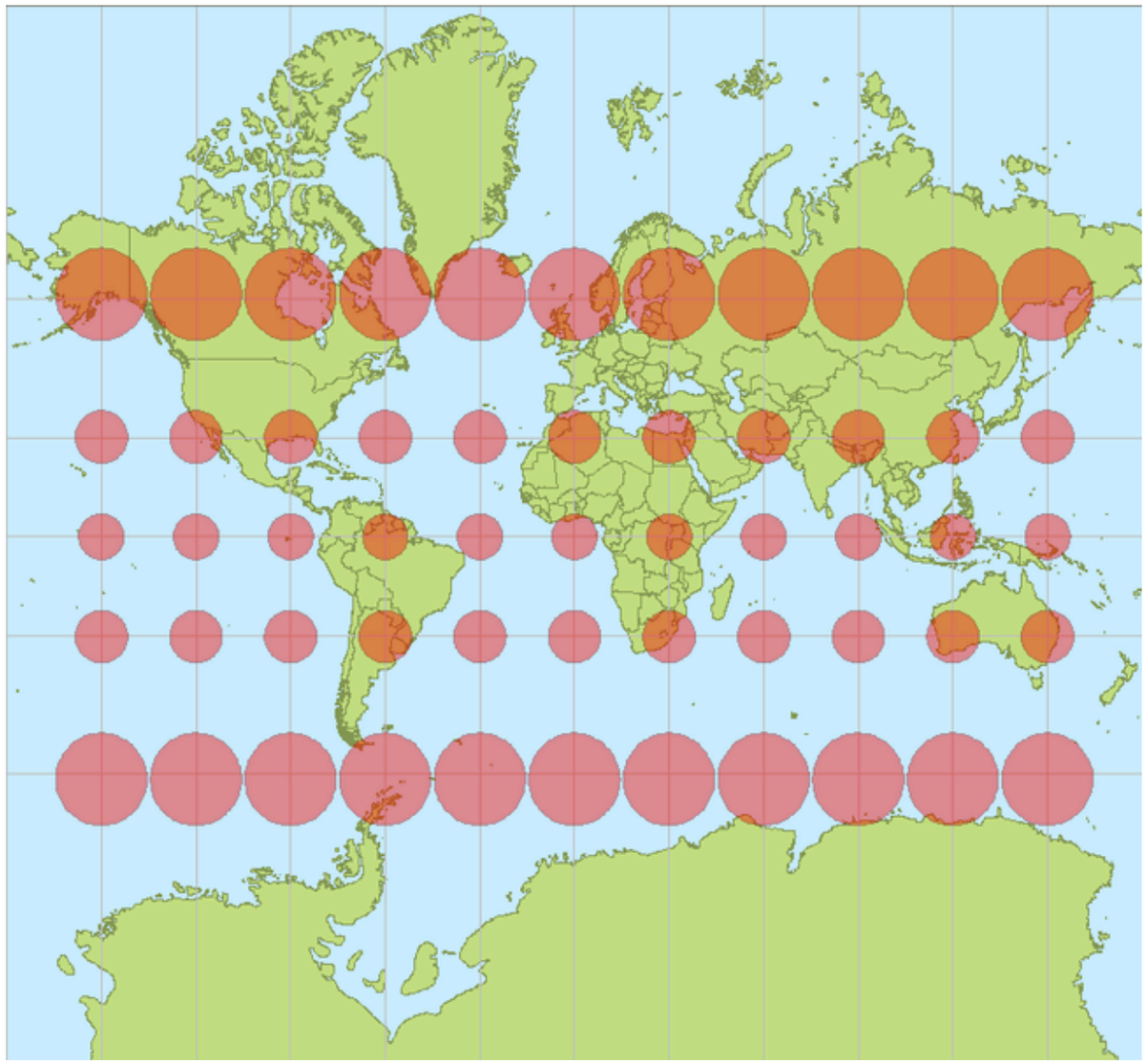
- The poles move “slower” than the equator
- Therefore: they look wider
- The map would be 3.14 longer than its height
- Solution: compensate the distortion by using a logarithmic “lens”, making the wide parts tall as well

The Mercator Map

Distortion Part II.

Because top parts are both wider and taller:

- Gronland seems as large as Africa (14 times smaller)
- Finland is as wide as India
- Result: twice the resolution of Helsinki than that of Kuala Lumpur

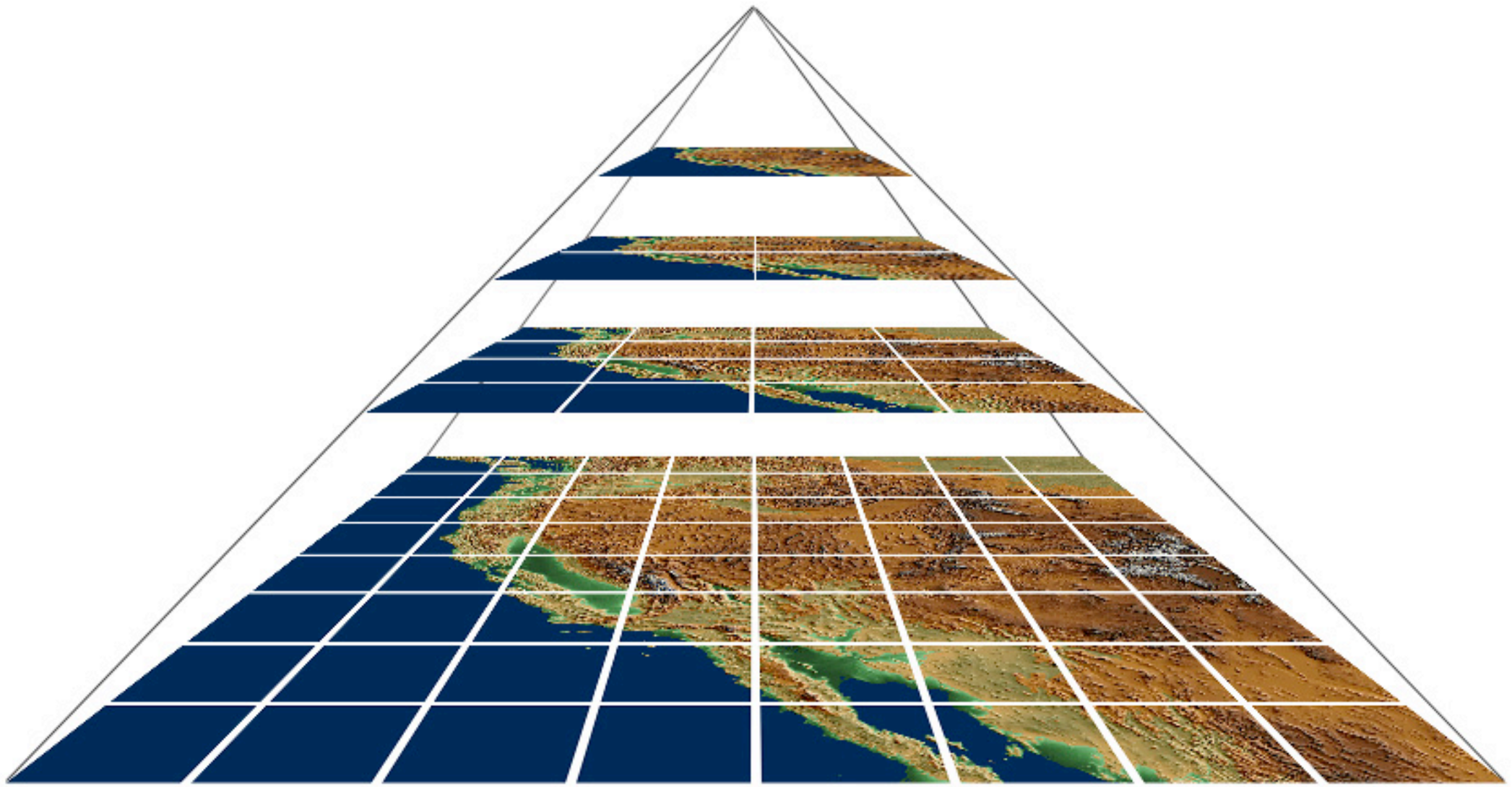


What are the benefits?

- It is universally applicable
- Directions remain the same (what's North, is displayed North)
- On small distances, the distortion is balanced
- It's square-shaped, square is nice

Every single webmap is mercator-based on every zoomlevel

Don’t rely on them when planning your trip from Helsinki to Kuala Lumpur



Part II: Computer Maps

Size problems

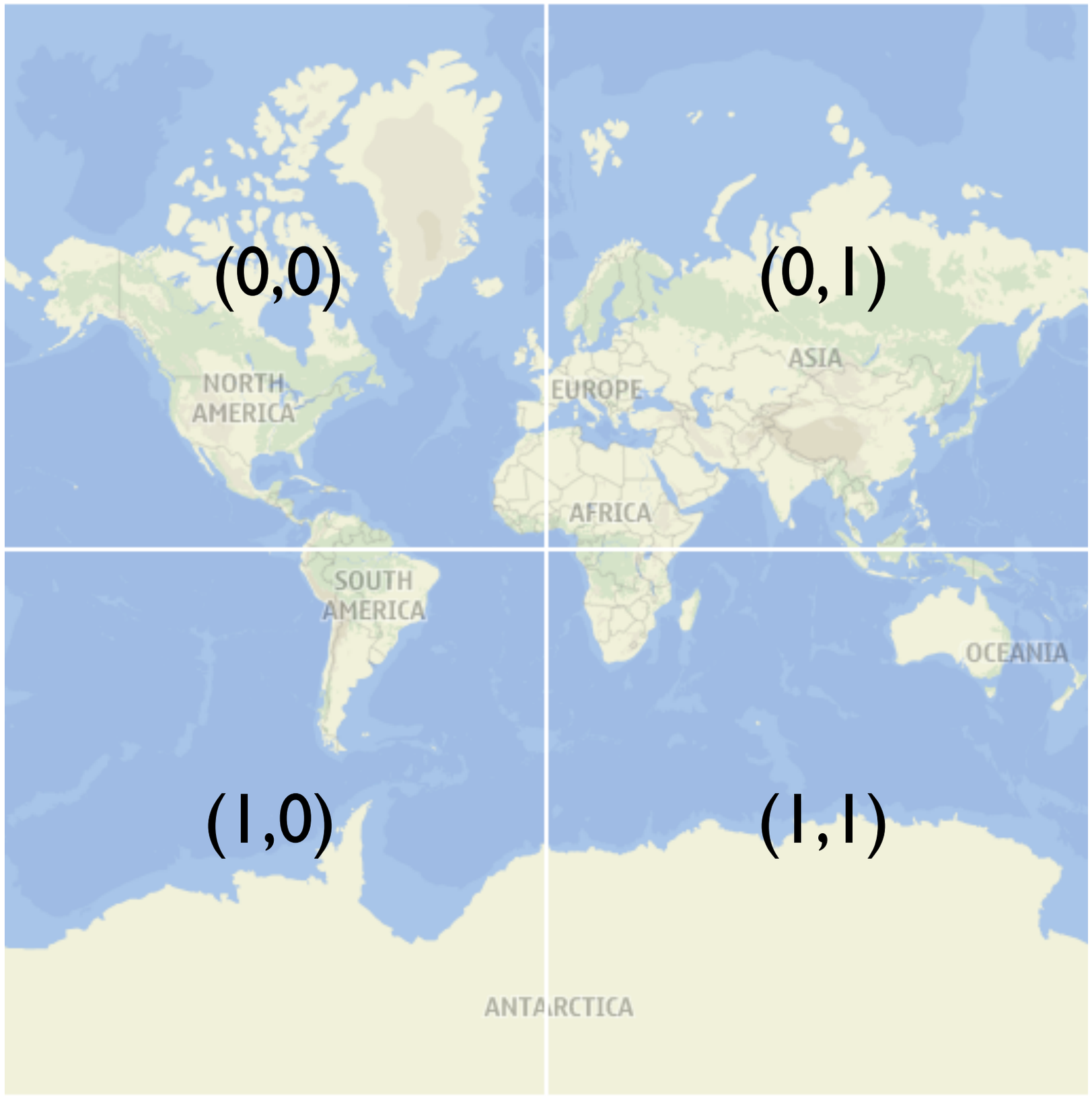
- Apple Macintosh, 1984
- 128 K RAM, black-and-white screen
- Black-and-white map of Budapest: 10.24 megabytes
- Likely won't fit into memory...

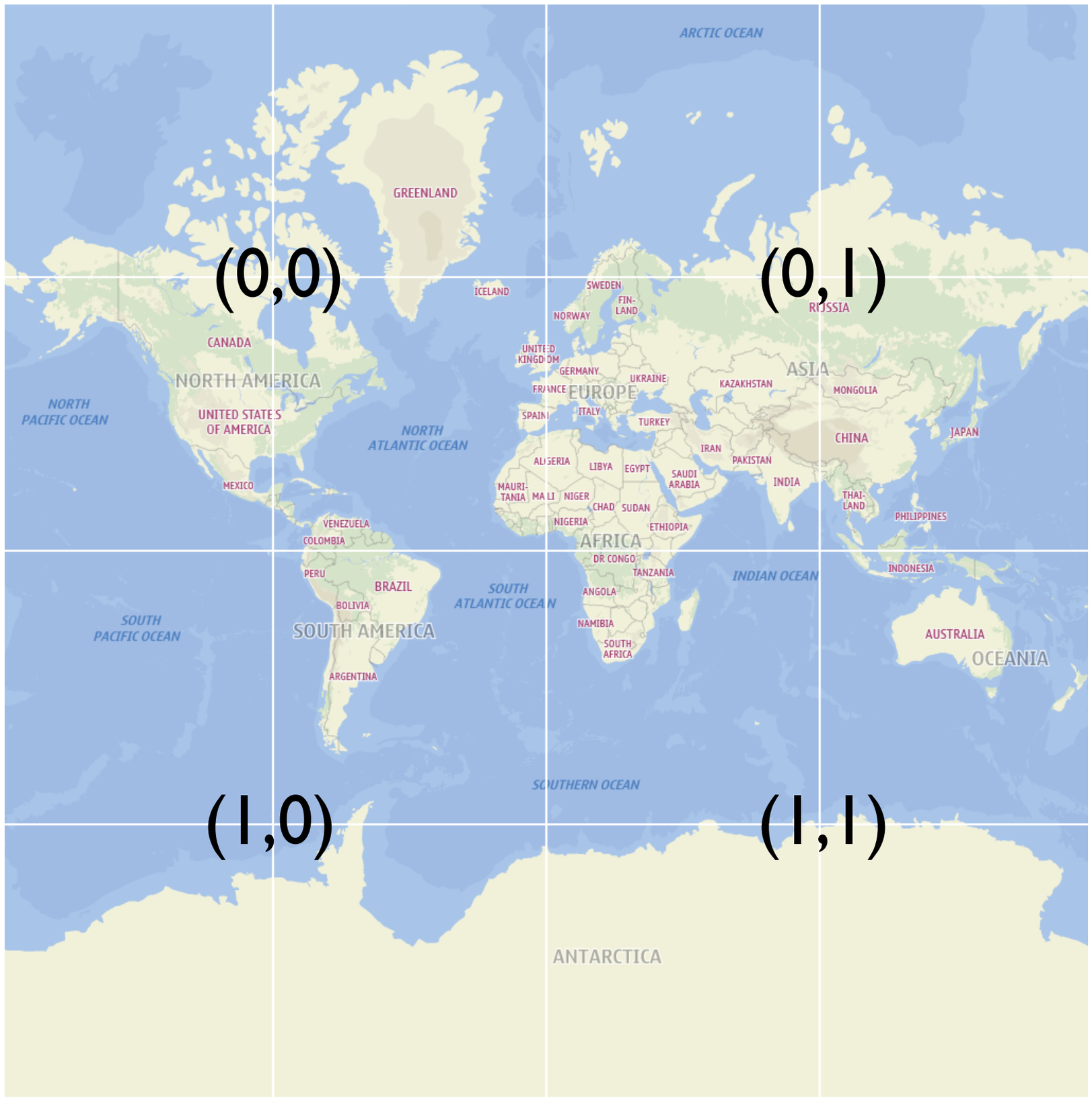
Divide and Conquer

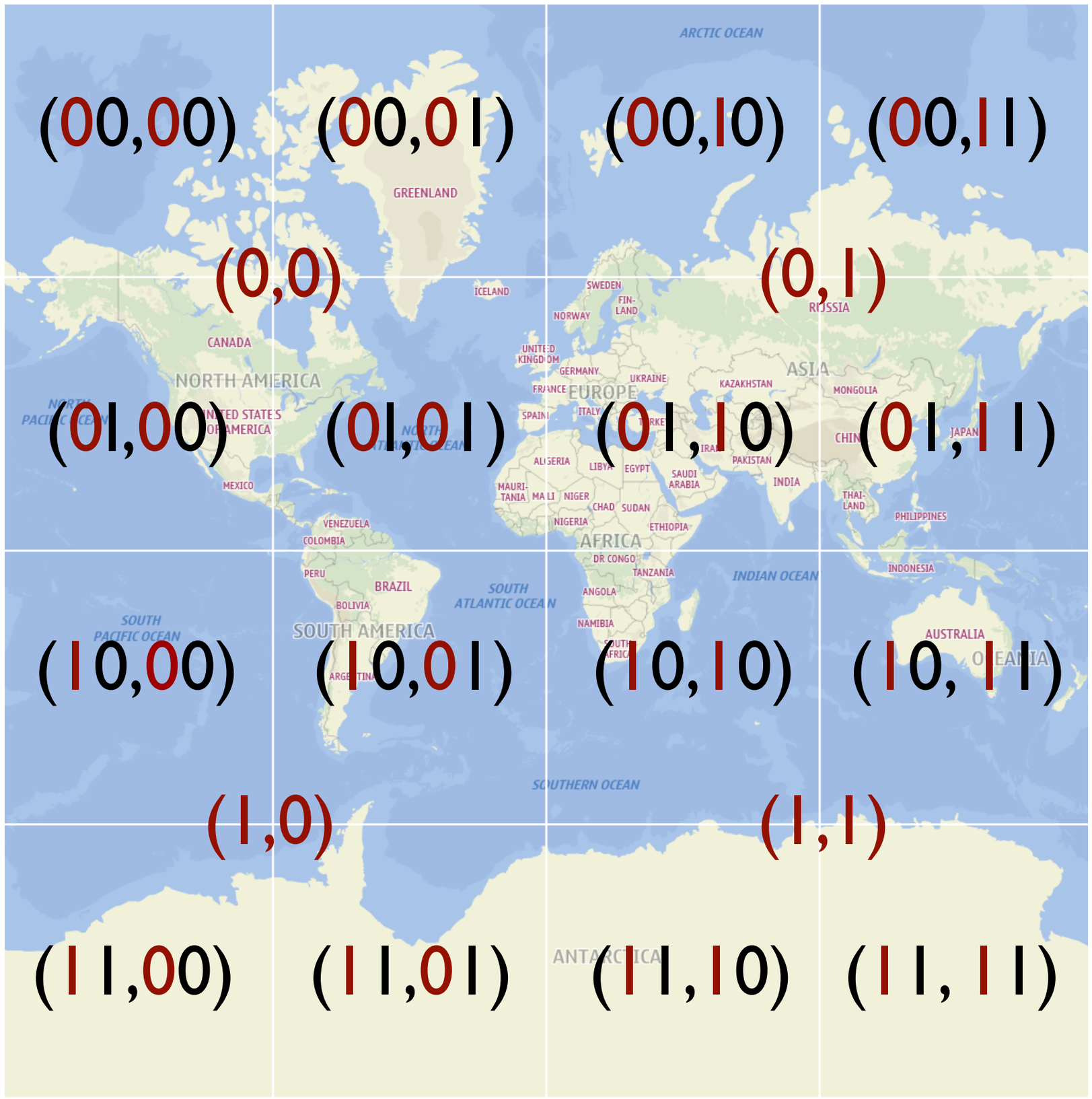
We are computer people, we think in binary...

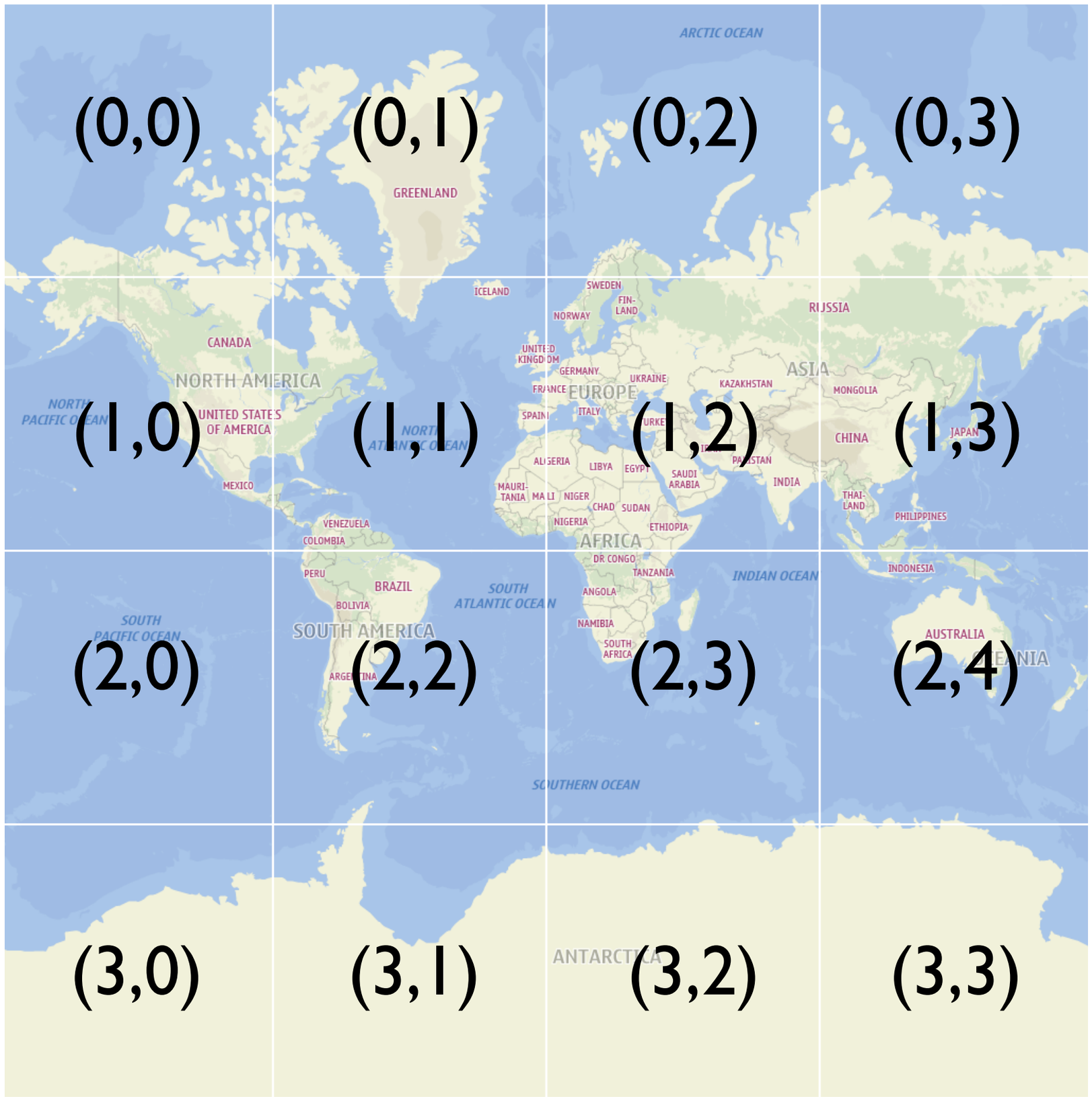

NORTH AMERICA
EUROPE
ASIA
AFRICA
SOUTH AMERICA
OCEANIA
ANTARCTICA

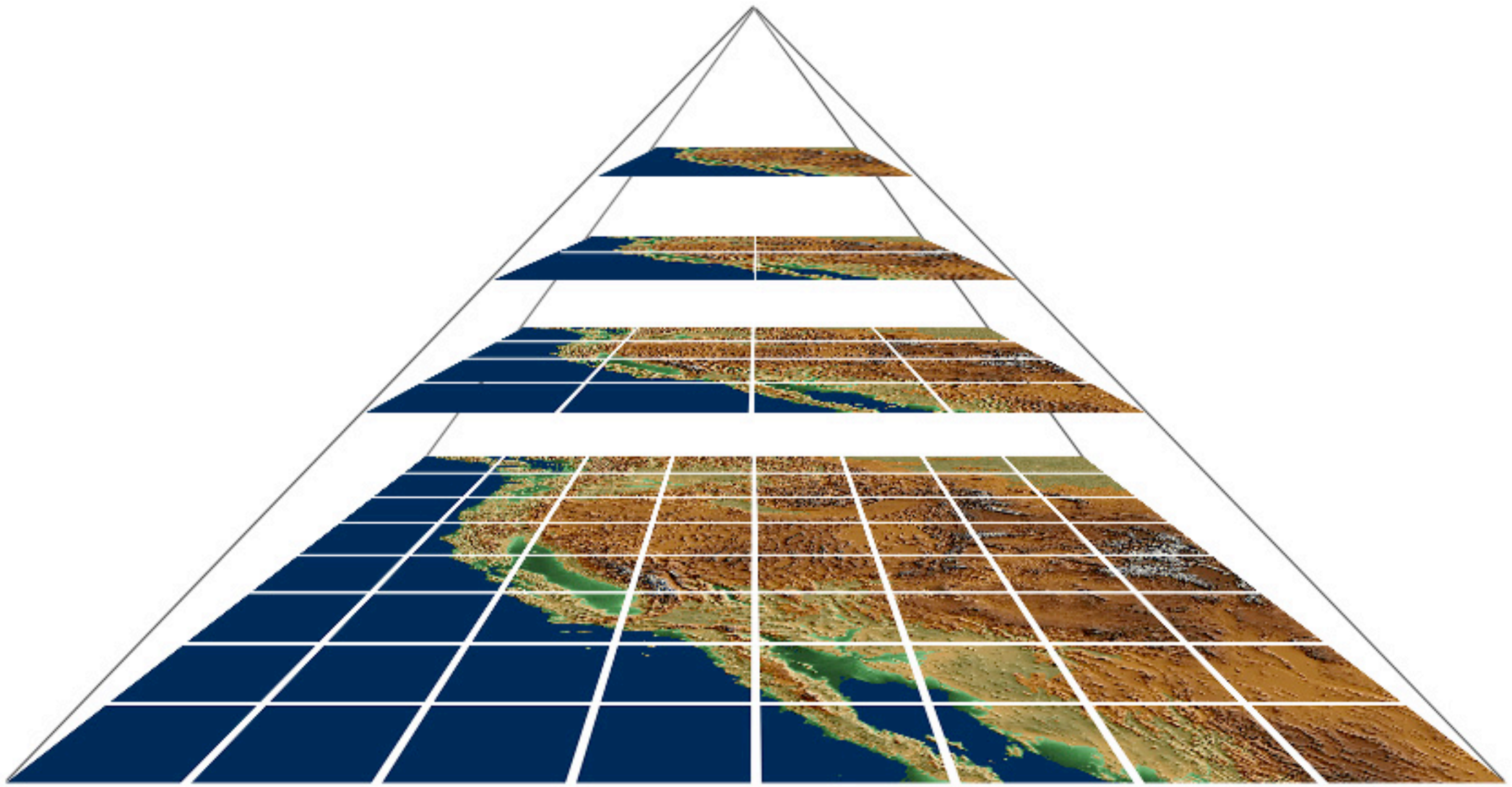

NORTH
AMERICA
EUROPE
ASIA
AFRICA
SOUTH
AMERICA
OCEANIA
ANTARCTICA


(0,0)
(0,1)
(1,0)
(1,1)
NORTH AMERICA
EUROPE
ASIA
AFRICA
SOUTH AMERICA
OCEANIA
ANTARCTICA


(0,0)
(0,1)
(1,0)
(1,1)
ARCTIC OCEAN
GREENLAND
ICELAND
SWEDEN
FIN-
LAND
NORWAY
RUSSIA
CANADA
NORTH AMERICA
UNITED KINGDOM
GERMANY
UKRAINE
KAZAKHSTAN
ASIA
MONGOLIA
FRANCE
EUROPE
NORTH
PACIFIC OCEAN
UNITED STATES
OF AMERICA
SPAIN
ITALY
TURKEY
NORTH
ATLANTIC OCEAN
CHINA
JAPAN
IRAN
ALGERIA
LIBYA
EGYPT
PAKISTAN
SAUDI
ARABIA
INDIA
MEXICO
MAURI-
TANIA
MALI
NIGER
CHAD
SUDAN
THAI-
LAND
PHILIPPINES
VENEZUELA
NIGERIA
ETHIOPIA
COLOMBIA
AFRICA
DR CONGO
TANZANIA
INDONESIA
INDIAN OCEAN
PERU
BRAZIL
SOUTH
ATLANTIC OCEAN
ANGOLA
BOLIVIA
SOUTH
PACIFIC OCEAN
SOUTH AMERICA
NAMIBIA
AUSTRALIA
SOUTH
AFRICA
OCEANIA
ARGENTINA
SOUTHERN OCEAN
ANTARCTICA


(00,00)
(00,01)
(00,10)
(00,11)
(0,0)
(0,1)
(01,00)
(01,01)
(01,10)
(01,11)
(10,00)
(10,01)
(10,10)
(10,11)
(1,0)
(1,1)
(11,00)
(11,01)
(11,10)
(11,11)
ARCTIC OCEAN
GREENLAND
ICELAND
SWEDEN
FIN-LAND
NORWAY
RUSSIA
CANADA
NORTH AMERICA
UNITED KINGDOM
GERMANY
UKRAINE
KAZAKHSTAN
ASIA
MONGOLIA
FRANCE
EUROPE
ITALY
SPAIN
JAPAN
CHINA
IRAN
PAKISTAN
ALGERIA
LIBYA
EGYPT
SAUDI ARABIA
INDIA
MEXICO
MAURI-TANIA
MALI
NIGER
CHAD
SUDAN
THAI-LAND
PHILIPPINES
VENEZUELA
COLOMBIA
NIGERIA
ETHIOPIA
AFRICA
DR CONGO
TANZANIA
INDIAN OCEAN
INDONESIA
PERU
BRAZIL
SOUTH ATLANTIC OCEAN
ANGOLA
BOLIVIA
SOUTH PACIFIC OCEAN
SOUTH AMERICA
NAMIBIA
AUSTRALIA
OCEANIA
SOUTHERN OCEAN
ANTARCTICA


(0,0)
(0,1)
(0,2)
(0,3)
(1,0)
(1,1)
(1,2)
(1,3)
(2,0)
(2,2)
(2,3)
(2,4)
(3,0)
(3,1)
(3,2)
(3,3)
ARCTIC OCEAN
GREENLAND
ICELAND
SWEDEN
FIN-LAND
NORWAY
RUSSIA
CANADA
NORTH AMERICA
UNITED KINGDOM
GERMANY
UKRAINE
KAZAKHSTAN
ASIA
MONGOLIA
FRANCE
EUROPE
SPAIN
ITALY
TURKEY
UNITED STATES OF AMERICA
NORTH PACIFIC OCEAN
NORTH ATLANTIC OCEAN
CHINA
JAPAN
ALGERIA
LIBYA
EGYPT
SAUDI ARABIA
INDIA
MEXICO
MAURI-TANIA
MALI
NIGER
CHAD
SUDAN
THAI-LAND
PHILIPPINES
VENEZUELA
COLOMBIA
NIGERIA
ETHIOPIA
AFRICA
DR CONGO
TANZANIA
INDIAN OCEAN
INDONESIA
PERU
BRAZIL
SOUTH ATLANTIC OCEAN
ANGOLA
BOLIVIA
SOUTH PACIFIC OCEAN
SOUTH AMERICA
NAMIBIA
SOUTH AFRICA
AUSTRALIA
OCEANIA
ARGENTINA
SOUTHERN OCEAN
ANTARCTICA



The Tile Pyramid

How many tiles will we have?

How many tiles will we have?

- If we don't zoom at all: 1 x 1
- If we zoom once: 2 x 2
- If we zoom twice: 4 x 4
- If we zoom 15 times?
- If we zoom n times?

How many tiles will we have?

- If we don't zoom at all: 1 x 1
- If we zoom once: 2 x 2
- If we zoom twice: 4 x 4
- If we zoom 15 times: 2^{15} x 2^{15}
- If we zoom n times: 2^n x 2^n

WMTS Coordinate of a tile

http://host.name/path/**n**/**row**/**column**

A tile is usually
256x256 pixels

Convention, practical

Trivia

- How large is the area covered by a single pixel on zoomlevel *15* in *Budapest*?

- How large is the area of a single pixel in *Rome*?

- How large is the area of a single pixel in *Helsinki*?

Trivia

- What is the area covered by a single pixel on zoomlevel *15* in *Budapest*?
- What is the area of a single pixel in *Rome*?
- What is the area of a single pixel in *Helsinki*?

- Budapest is on *47.5°* (approx)
- Earth's circumference is *40 075 017* meters
- Every longitudinal line is of equal length with mercator projection
- Their real length is cos(*degree*)*equator

Trivia

- What is the area covered by a single pixel on zoomlevel *15* in *Budapest*?

- What is the area of a single pixel in *Rome*?

- What is the area of a single pixel in *Helsinki*?

- On zoomlevel 15 a longitudinal line is *8388608* pixels long ($2^{15}*256 = 2^{23}$)

- 40 075 017 * cos(47.5°)/ 8 388 608 ≈ 3.23 meters / pixel horizontal

- Vertical made the same by the logarithm

Trivia

- What is the area covered by a single pixel on zoomlevel *15* in *Budapest*?

- What is the area of a single pixel in *Rome*?

- What is the area of a single pixel in *Helsinki*?

- *Budapest*: 3.23 m - 10 m^2/pixel

- *Rome*: 3.56 m - 12.64 m^2/pixel

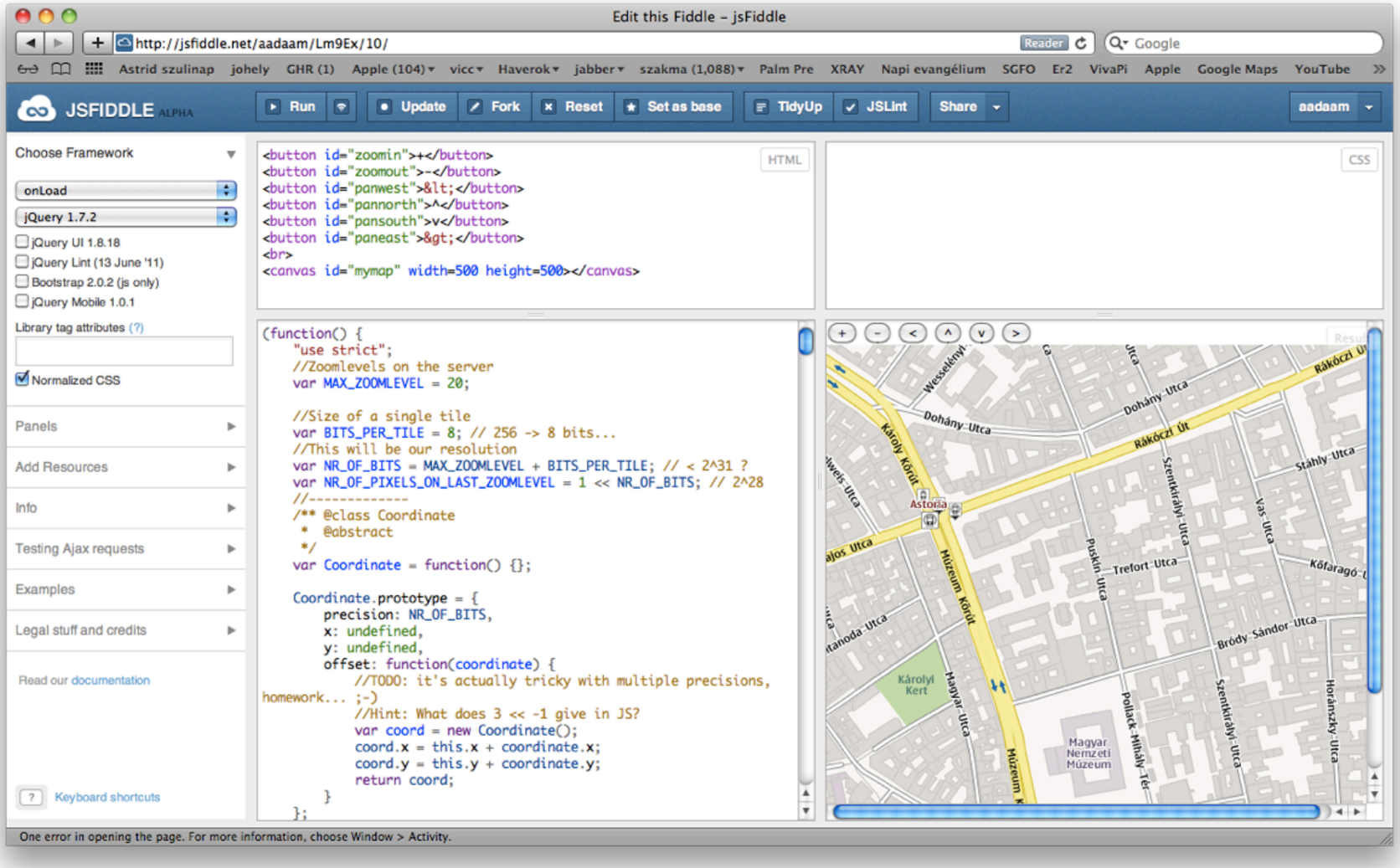
- *Helsinki*: 2.38 m - 5.64 m^2/pixel

- *Kuala Lumpur*: 4.77 m - 22 m^2/pixel

Trivia

- What is the area covered by a single pixel on max *(20)* Google / Nokia zoomlevel?

- *Budapest*: 10 cm - 100 cm^2/pixel
- *Rome*: 11.11 cm - 123 cm^2/pixel
- *Helsinki*: 7.4 cm - 55 cm^2/pixel
- *Kuala Lumpur*: 14 cm - 222 cm^2/pixel



http://jsfiddle.net/aadaam/Lm9Ex/ 0-10 /

Part III: Build your own map

Basic considerations

- Display a map at any given coordinate and zoomlevel
- Be able to pan around and zoom in&out
- Make it generic so knowledge could be used with other technologies as well
- Don’t deal with bugs

Basic considerations

- Display a map at any given coordinate and zoomlevel
- Be able to pan around and zoom in&out
- Make it generic so knowledge could be used with other technologies as well - **canvas**
- Don’t deal with bugs - **single browser**...

Step 1: display a single tile

http://jsfiddle.net/aadaam/Lm9Ex/0/

Step 2: Calculate your position

- Normalized spherical Mercator function: converts (latitude, longitude) to (x,y) between ((0..1),(0..1))
- Doesn't cover the poles well
- It's a real number - we prefer integers

Step 2: Calculate your position

- We will convert to Mercator once and don't deal with it afterwards
- JS uses `double`: precision up until 2^{53}
  PHP int precision: 2^{31}
- Theoretical resolution on 2^{31} for Budapest:
  1.38 cm / pixel - good enough
- We will use 2^{28} now, as our last zoomlevel is 20, tile is 256 wide

Step 3: Calculate your position's tile

- Where are we standing right now on zoomlevel 15?
- This office is at 47.49658, 19.057811 according to Google

- That at (55.29383638888884%, 44.605757935735435%) of the map
- Which is at (148428262, 119737670) of our 28-bit representation

Step 3: Calculate your position's tile

- Which is at (148428262, 119737670) of our 28-bit representation
- What does it take to go from a 28-element representation to a 15-element representation?

Step 3: Calculate your position's tile

- Which is at (148428262, 119737670) of our 28-bit representation
- What does it take to go from a 28-element representation to a 15-element representation?
- `x >> (28 - 15)`
  `y >> (28 - 15)`

2

Zooming in

- Store your current zoomlevel into a variable
- Re-calculate tile coordinates based on this variable after each movement (panning, zooming)
- Allow this variable to be overwritten

Panning

- Choose a reference point
- Usual practice is to choose the map center
- We'll use topleft corner instead (easier to calculate with)

The three coordinate systems

Reference coordinate system

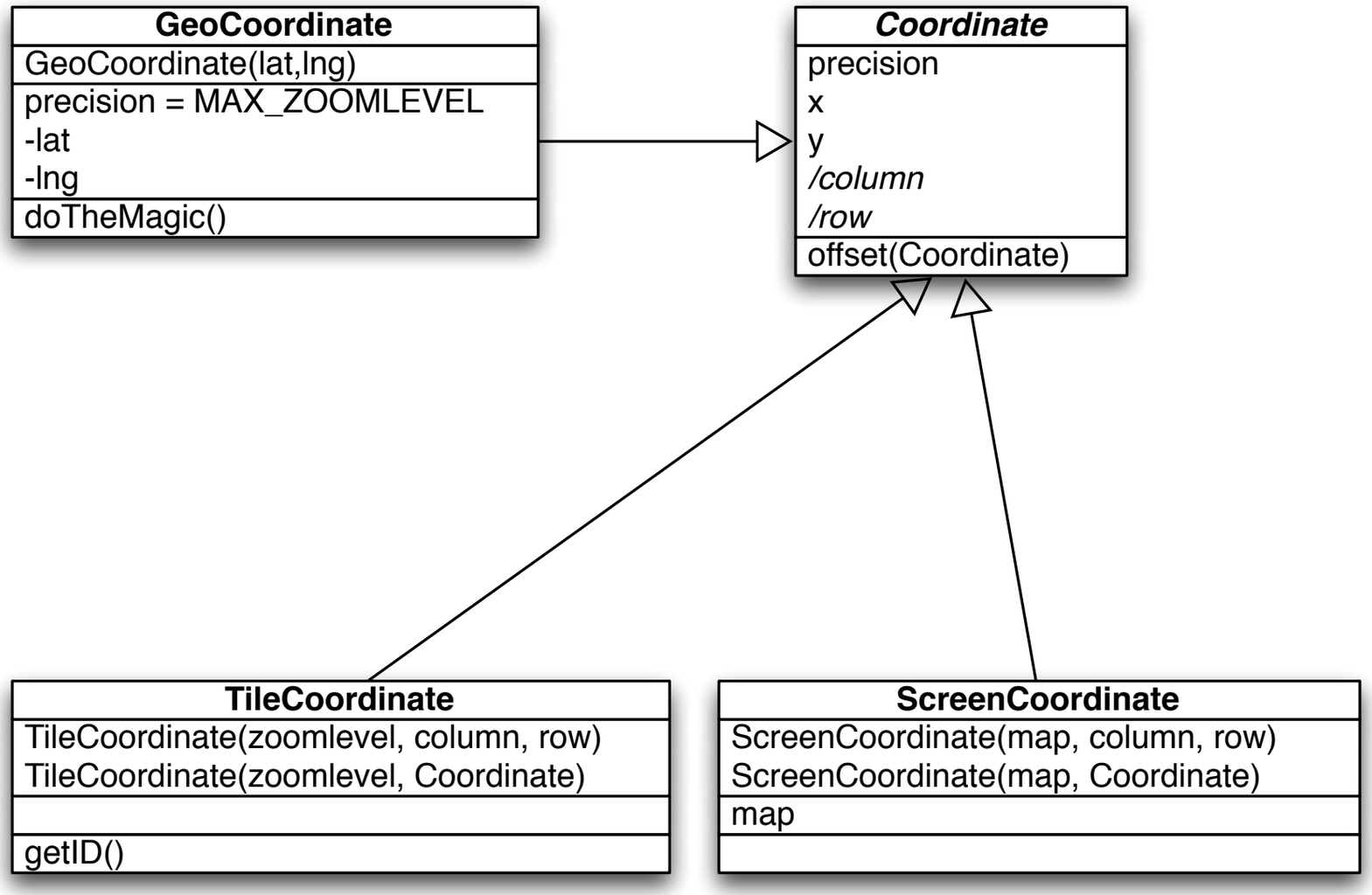
- 28-bit
- absolute
- deals with conversion between globe and mercator

Tile coordinate system

- 15-bit
- zoomlevel-dependent

Screen coordinate-system

- Like tile coordinate system
- 15-bit + 8-bit
- zoomlevel-dependent
- has an offset

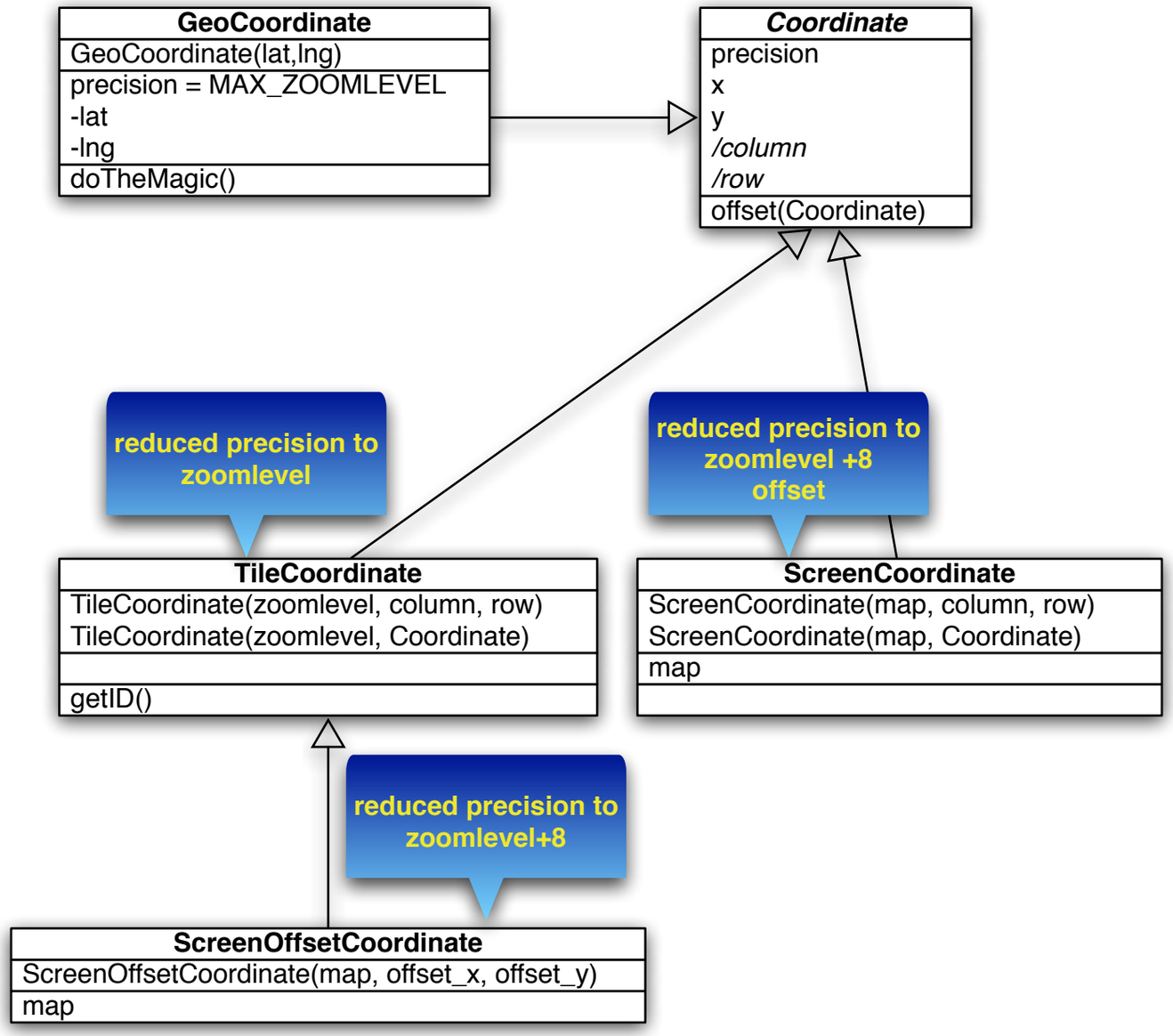


The three coordinate systems

ES 5 Magic

- Calculate coordinates from underlying absolute coordinate dynamically through getters and setters

- Isn't really needed as coordinates are better if non-mutable and throw-away
- Another mathematical solution would be to use Monads



It's worth introducing a fourth one as well

How to display multiple tiles

- We know the topleft coordinate - in any coordinate system
- We know our width and height - in ScreenOffset coordinate system
- We know what zoomlevel we are at

How to display multiple tiles

1. Get the tile coordinate of the topleft pixel ($z, x1, y1$)
2. Get the tile coordinate of the bottomright pixel ($z, x2, y2$)
3. Request all the tiles on z between $x1, x2$ and $y1, y2$

How to display multiple tiles

4. When the tile image arrives, send the tile coordinate with it as well.
5. Get the screen coordinate of the topleft corner of the tile by *shifting* it with tilesize in bits (Why will this work?), minus topleft
6. Display the tile
7. Make sure you don't allow late callbacks after zoom

Bringing simple interaction

- On user testing, ~80% of people used panning cursors instead of mouse gestures
- There's a nice track on tileserver statistics across the Atlantic
- Never scale down tiles for permanent display!

Cartographers are allergic to this!

(sorry for the scaledowns explaining the pyramid...)

Bringing simple interaction

- Let's have some buttons (non-overlay)
- zoom in:
    1. increase zoomlevel of map
    2. re-render
- pan north/south/top/down
    - add/substract 15 screenoffset coordinates from the topleft corner respectively
    - re-render

ES5 Magic #2

- Pattern: when panning or zoomlevel is modified, we have to re-render
- re-render automatically
- Do this only if the map was already rendered once
- Don’t forget to cancel current downloads on zoomlevel change!

Pan on mouse drag

- Usually, there is kinetic movement after - we don't care about this
- Only offset difference is needed - pageX will do

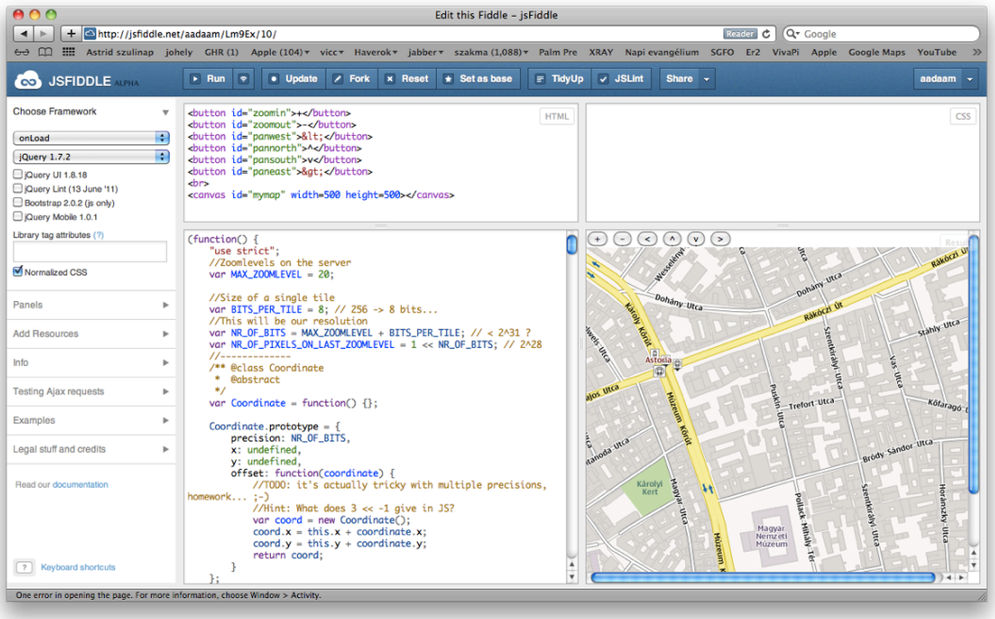
1. Remember offset at mousedown
2. Cancel drag mode on mouseout or mouseup
3. On mousemove, calculate offset difference, pan, remember new offset

Zoom to location by click

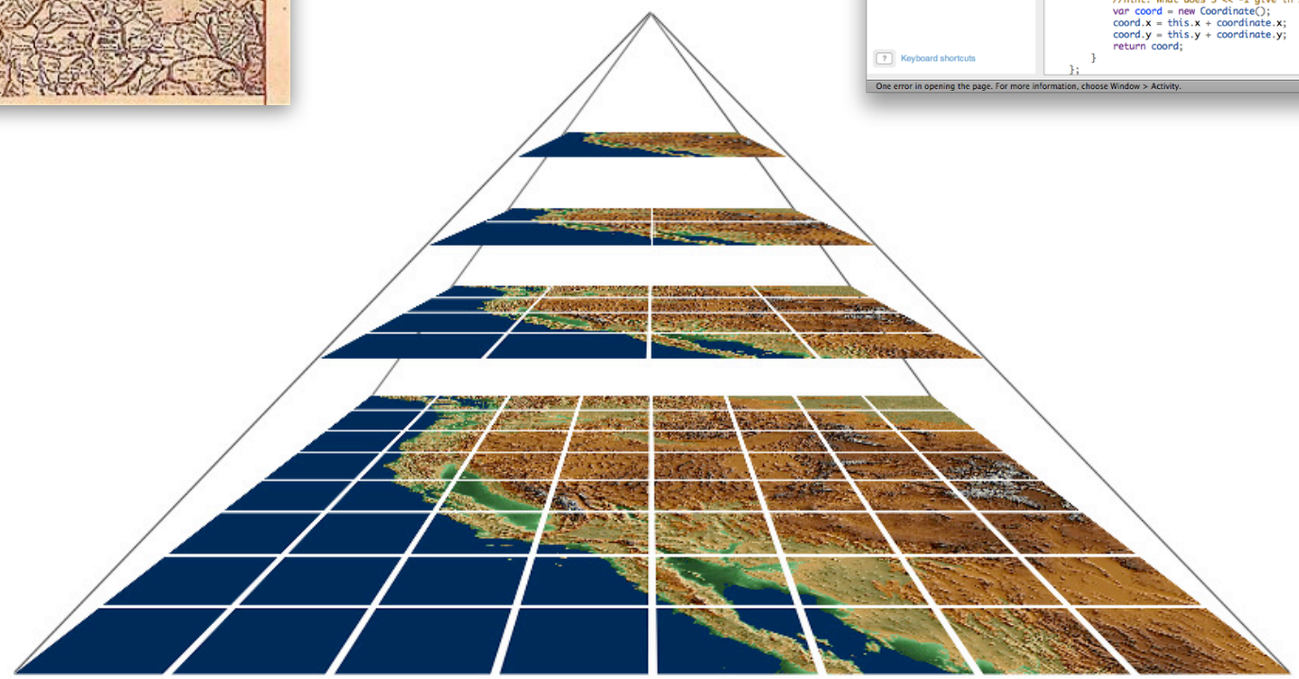
- You'll have two ScreenCoordinate systems: one at the old, one at the new zoomlevel.
- Aim: ScreenCoordinate(ev.offset) == ScreenCoordinate'(ev.offset)
- **On zoom-in:** `topleft+=1/2 * ScreenOffset(ev.offset)`
- **On zoom-out:** `topleft-= -1 * ScreenOffset(ev.offset)`

Add caching

- We shouldn't create Image objects on every render
- Also, we should cancel current downloads if user has zoomed away
- We shouldn't download while panning either


Edit this Fiddle - jsFiddle
http://jsfiddle.net/aadaam/Lm9Ex/10/
JSFIDDLE ALPHA
Run
Update
Fork
Reset
Set as base
TidyUp
JSLint
Share
aadaam
Choose Framework
onLoad
jQuery 1.7.2
jQuery UI 1.8.18
jQuery Lint (13 June '11)
Bootstrap 2.0.2 (js only)
jQuery Mobile 1.0.1
Library tag attributes (?)
Normalized CSS
Panels
Add Resources
Info
Testing Ajax requests
Examples
Legal stuff and credits
Read our documentation
Keyboard shortcuts
HTML
CSS
<button id="zoomin">++</button>
<button id="zoomout">--</button>
<button id="panwest"><</button>
<button id="pannorth">^</button>
<button id="pansouth">v</button>
<button id="paneast">></button>

<canvas id="mymap" width=500 height=500></canvas>
(function() {
"use strict";
//Zoomlevels on the server
var MAX_ZOOMLEVEL = 20;
//Size of a single tile
var BITS_PER_TILE = 8; // 256 -> 8 bits...
//This will be our resolution
var NR_OF_BITS = MAX_ZOOMLEVEL + BITS_PER_TILE; // < 2^31 ?
var NR_OF_PIXELS_ON_LAST_ZOOMLEVEL = 1 << NR_OF_BITS; // 2^28
//-------------
/** @class Coordinate
* @abstract
*/
var Coordinate = function() {};
Coordinate.prototype = {
precision: NR_OF_BITS,
x: undefined,
y: undefined,
offset: function(coordinate) {
//TODO: it's actually tricky with multiple precisions, homework... ;-)
//Hint: What does 3 << -1 give in JS?
var coord = new Coordinate();
coord.x = this.x + coordinate.x;
coord.y = this.y + coordinate.y;
return coord;
}
};
One error in opening the page. For more information, choose Window > Activity.



Wrap-up

Remember

- Store GeoCoords on high-precision grid systems instead of floats (it's nasty when they're serialized into strings back-and-forth...), but beware of the dateline!
- Zooming equals shifting
- You can calculate short distances on quasi-equal longitudinal spots with cos(latitude), don't do it for large distances
- All maps are flat! While WebGL is changing this, satellite will stay as such for a while

URLs

- http://jo-hely.hu/~aadaam/minimap/ - original implementation
- http://api.maps.nokia.com http://m.maps.nokia.com
- Leaflet, in case you need an Open Source one
- Slides: http://www.slideshare.net/aadaam1
- GitHub: @aadaam

Thanks to

- *Astrid Fasold*, Nokia Maps Map Design
- *Andrea Giammarchi*, Nokia Maps HTML5 Mobile Maps
- *Thomas Fischer, Jörg Hösel*, Nokia Maps (Web) API
- *Aniko Szalay* (mug projection shot)

Acknowledgements

- Maps used in examples are from Nokia Maps - http://maps.nokia.com
- Mercator's map and globe are from Atlas Obscura - support them! http://atlasobscura.com/place/mercator-museum
- Tile pyramid was made by Michael Potmesil at Bell Labs- http://www.geckil.com/~harvest/www6/Technical/Paper130/Paper130.html
- The Tissot Indicatrix illustration about Mercator projection is from Wikipedia
- Project made tremendous use of JSFiddle

Any Questions?