



17: Performance





performance

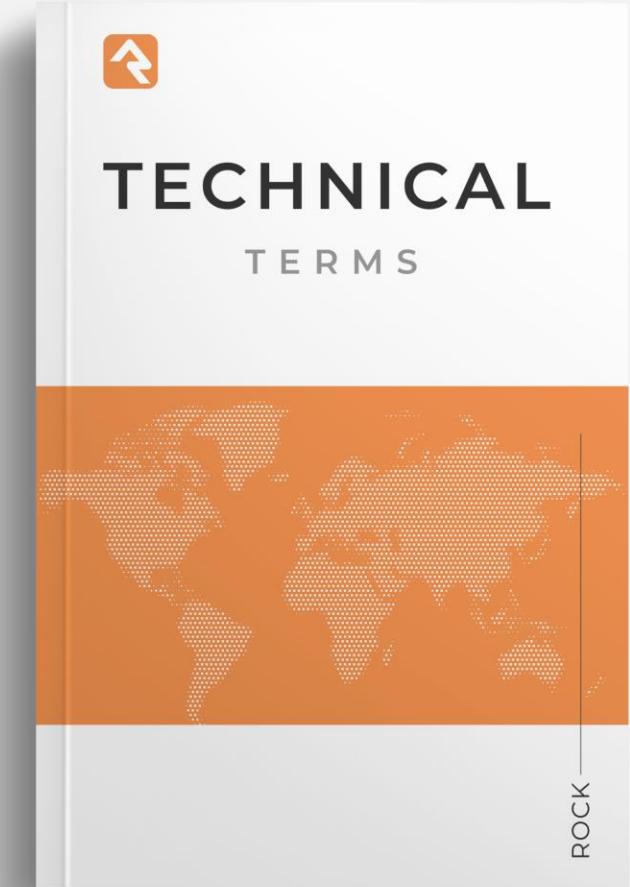
[per-form-uhns]

The amount of useful work
accomplished by a computer.



Demystified:

The “faster” a computer is, the more work it can do in the same amount of time.





performance

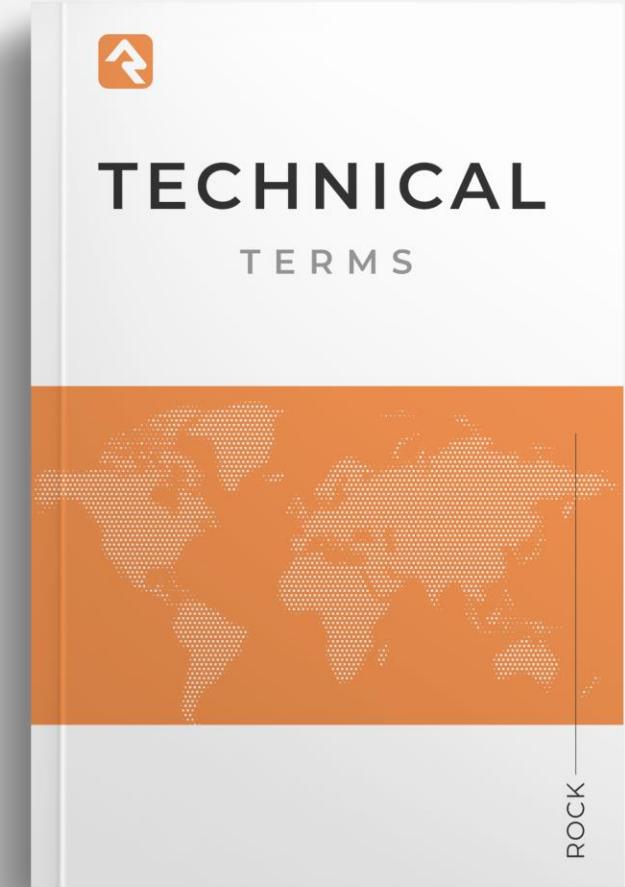
[per-form-uhns]

The amount of useful work
accomplished by a computer.



Demystified:

If you have a set amount of work that needs to be done, better performance allows it to complete sooner.





Websites that are too fast:

-
-
-





Performance is a Partnership

- Core System Efficiency
- Configuration Impact
 - (including server resources)
- Custom Development Impact

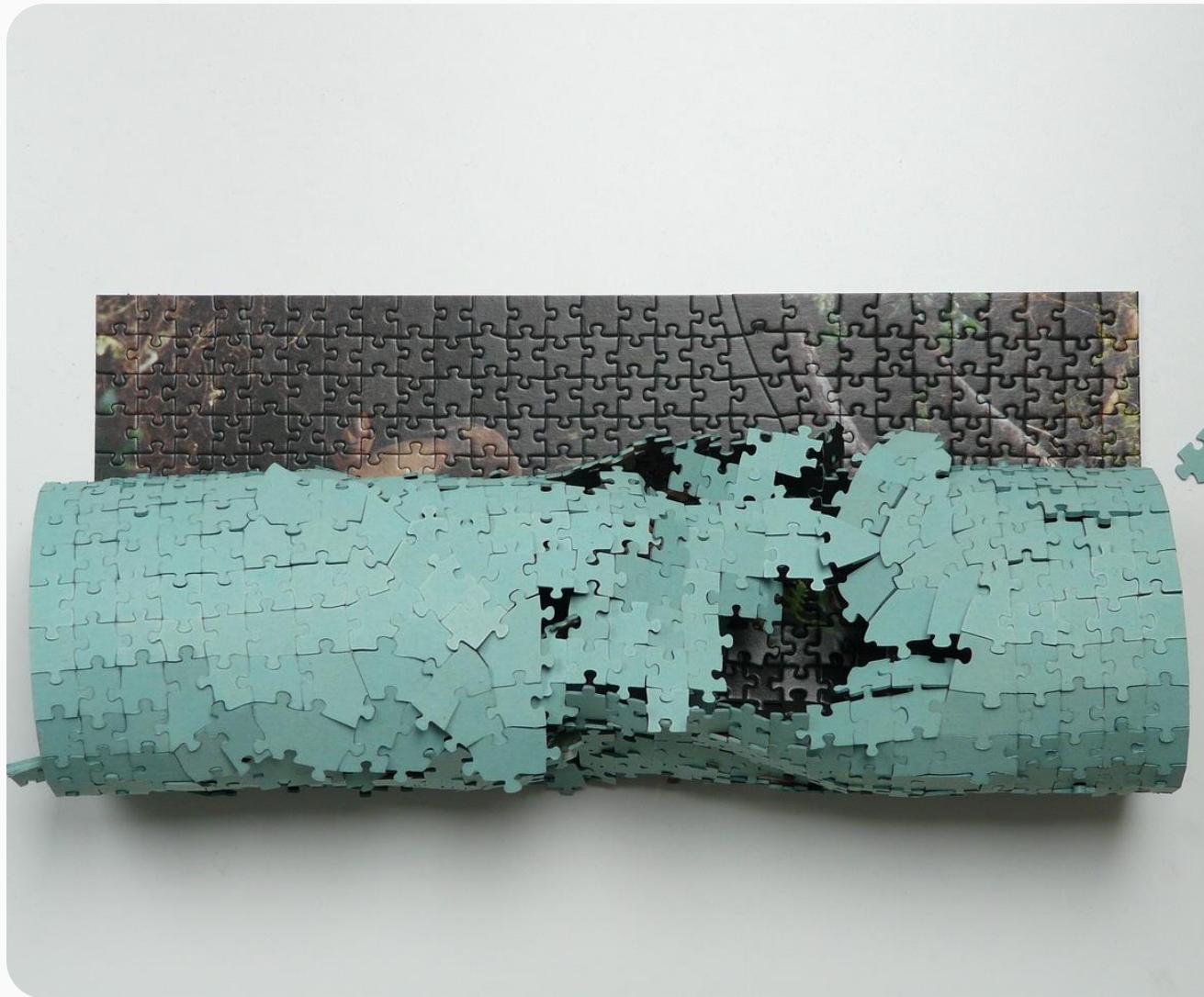




Rock's Performance Challenges:

1. Extensibility is hard.

- It's easy to make a page efficient when it only ever has to do one thing.
- But in Rock, each church can add content and configure any page.
- We are constantly improving efficiency, measuring in fractions of milliseconds.





Rock's Performance Challenges:

2. Usage is Concentrated.

- A server must have its resources set up before they're needed.
- Peak usage isn't evenly distributed through a day or week.
- Requirements change as your church and database grows.





Rock's Performance Challenges:

3. Customization is a double-edged blade.
 - If a church adds a single slow experience on a really fast page, the whole page suffers.
 - In these cases, the whole site feels slow to your visitors.





Performance is a Partnership

- The core team takes care of the core efficiency and provides guidelines on server resourcing.
- Let's talk about making sure your customizations are efficient.



Principle:

Rock is *fast*.
But Lava templates, if not carefully designed, can undo that.





When your pages are loaded, they need to get stored information.

- The server makes a request for the information and needs to wait to get the data.
- There are lots of places the information can be stored.





What requests are faster and slower than others?

- Computers are fast, so let's break things into human-scale.
- Assume one processor cycle is equivalent to you taking one step.





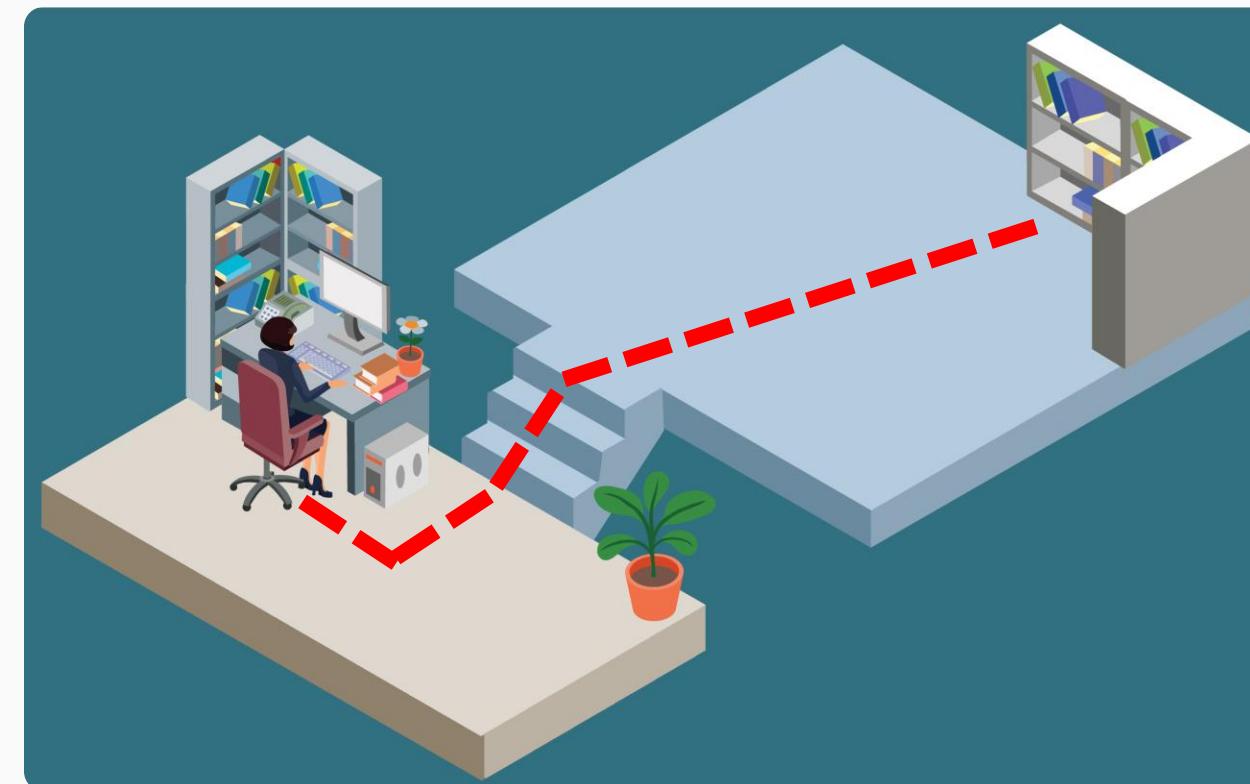
L1 Cache: 3 steps (7.5 feet)





L1 Cache: 3 steps (7.5 feet)

L2 Cache: 9 steps (22.5 feet)

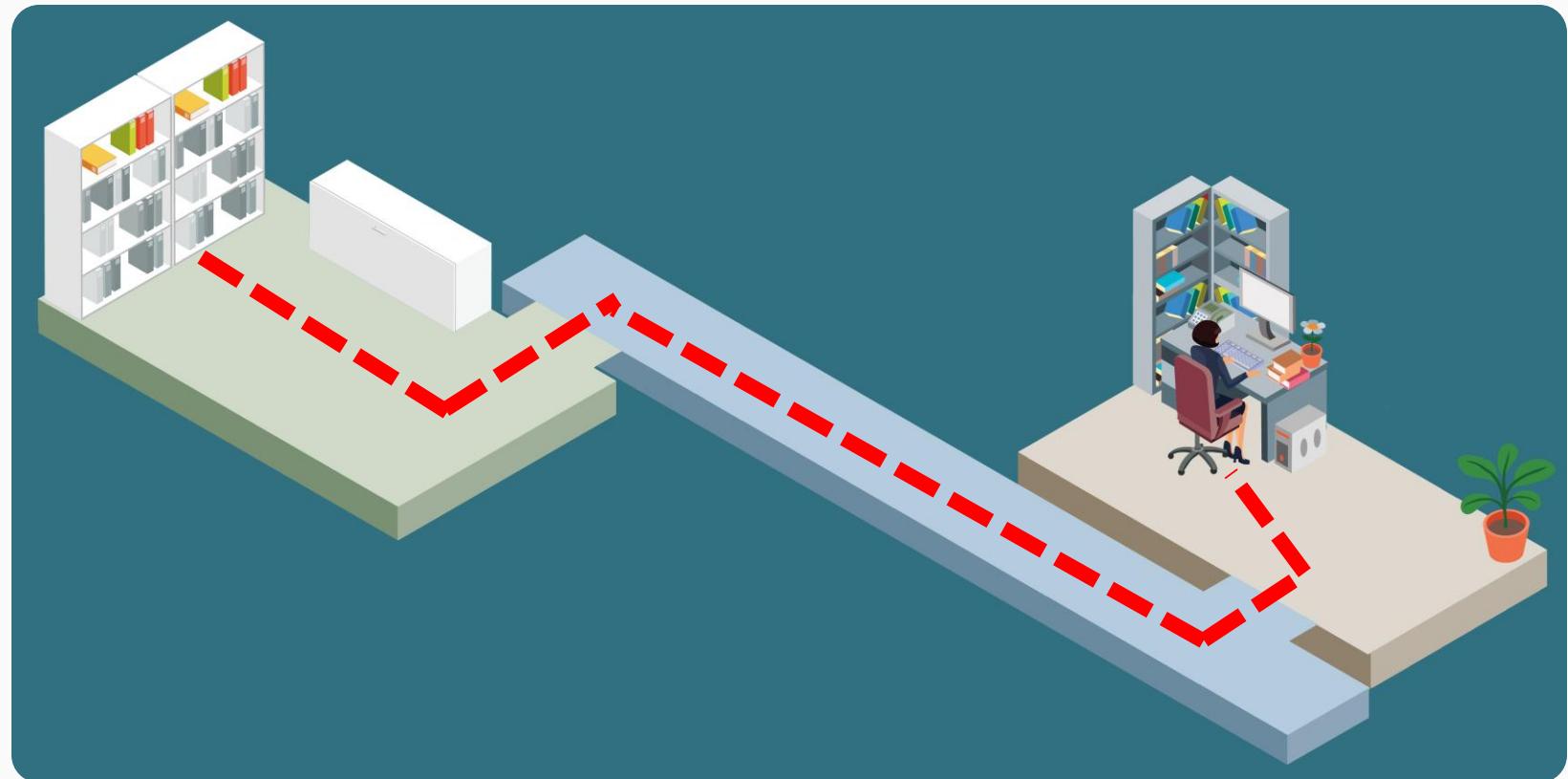




L1 Cache: 3 steps (7.5 feet)

L2 Cache: 9 steps (22.5 feet)

L3 Cache: 43 steps (107.5 feet)





L1 Cache: 3 steps (7.5')

L2 Cache: 9 steps (22.5')

L3 Cache: 43 steps (107.5')

Main Memory: 300 steps (750')





L1 Cache: 3 steps (7.5')

L2 Cache: 9 steps (22.5')

L3 Cache: 43 steps (107.5')

Main Memory: 300 steps (750')

SSD access: 150,000 steps (71mi)





L1 Cache: 3 steps (7.5')

L2 Cache: 9 steps (22.5')

L3 Cache: 43 steps (107.5')

Main Memory: 300 steps (750')

SSD access: 150,000 steps (71mi)

HDD access: 3 million steps
(1,420 mi)





L1 Cache: 3 steps (7.5')

L2 Cache: 9 steps (22.5')

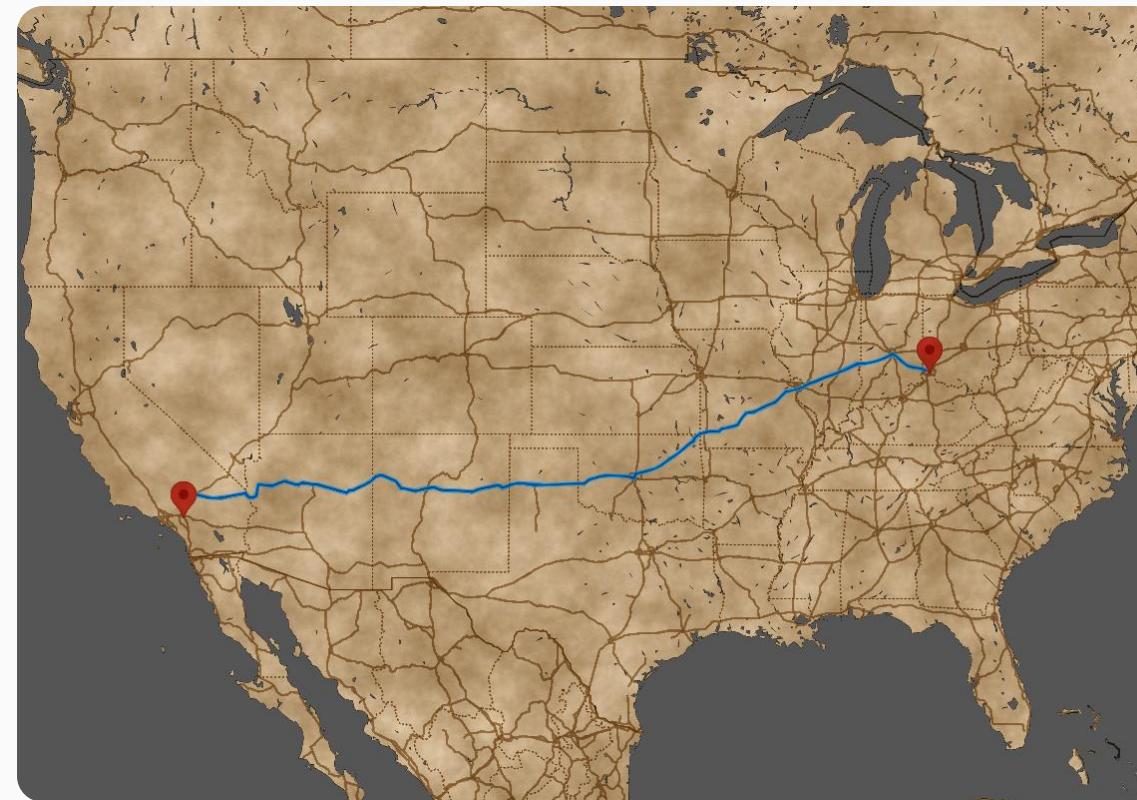
L3 Cache: 43 steps (107.5')

Main Memory: 300 steps (750')

SSD access: 150,000 steps (71mi)

HDD access: 3 million steps
(1,420 mi)

Simple Database Read: 7.75m
steps (3,700 miles)





L1 Cache: 3 steps (7.5')

L2 Cache: 9 steps (22.5')

L3 Cache: 43 steps (107.5')

Main Memory: 300 steps (750')

SSD access: 150,000 steps (71mi)

HDD access: 3 million steps
(1,420 mi)

Simple DB Read: 7.75m steps

Complex DB Read: 59.75m steps
(28,290 miles)





Clearly, it's better to use storage with faster retrieval.

Rock automatically stores certain records in cache (RAM)

- Defined Value/Types
- Campus
- Page
- Block
- Attribute
- (etc)





FromCache

```
1  {%- assign schoolList = 34 | FromCache:'DefinedType' %}
```

Model Details

Filter Options ▾

Enabled Filters
IsLava: True

Show: □ Methods

DefinedType

DefinedType Logic

Properties

Attributes

AttributeValues

CategorizedValuesEnabled Gets or sets a flag indicating if the Defined Values associated with this Defined Type can be grouped into categories.

Category Gets or sets the category.

CategoryId Gets or sets the category identifier.

CreatedByPersonAliasId

CreatedByPersonId

CreatedByPersonName

CreatedDateTime Gets or sets a collection containing the [DefinedValues](#) that belong to this DefinedType.

DefinedValues Gets or sets a user defined description of the DefinedType.

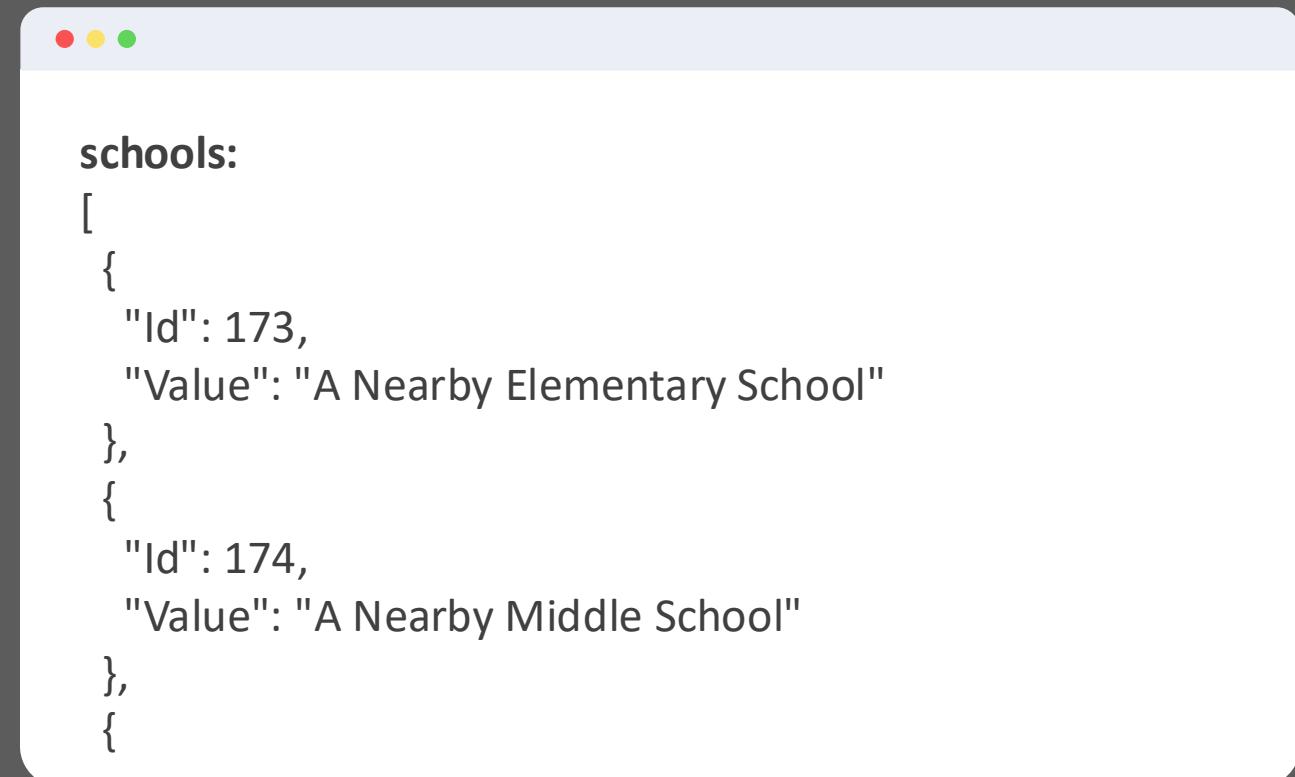
schoolList:

```
{  
  "Id": 34,  
  "Name": "School",  
  "Description": "Is where the person currently attends  
or last attended school."  
}
```



FromCache

```
1  {% assign schoolList = 34 | FromCache:'DefinedType' %}  
2  {% assign schools = schoolList.DefinedValues %}
```



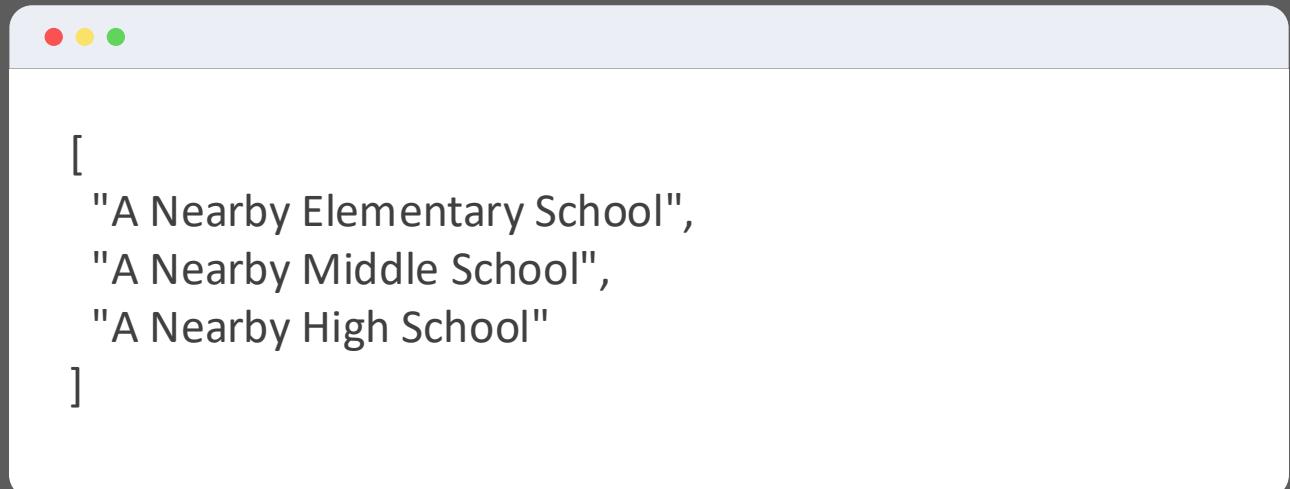
A screenshot of a Mac OS X application window titled "schools". The window contains the following JSON data:

```
schools:  
[  
  {  
    "Id": 173,  
    "Value": "A Nearby Elementary School"  
  },  
  {  
    "Id": 174,  
    "Value": "A Nearby Middle School"  
  },  
  {  
    "Id": 175,  
    "Value": "A Nearby High School"  
  }]
```



FromCache

```
1  {% assign schoolList = 34 | FromCache:'DefinedType' %}  
2  {% assign schools = schoolList.DefinedValues %}  
3  {{ schools | Map:'Value' | ToJSON }}
```



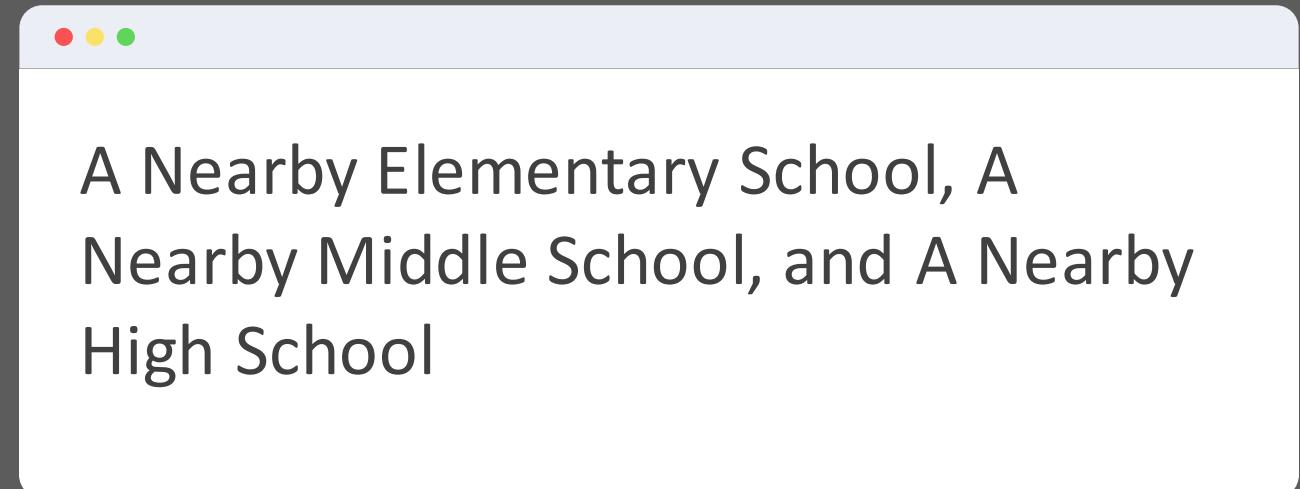
A screenshot of a Mac OS X application window showing a JSON array. The window has a white background and a title bar with red, yellow, and green buttons. The JSON array contains three strings: "A Nearby Elementary School", "A Nearby Middle School", and "A Nearby High School".

```
[  
  "A Nearby Elementary School",  
  "A Nearby Middle School",  
  "A Nearby High School"  
]
```



FromCache

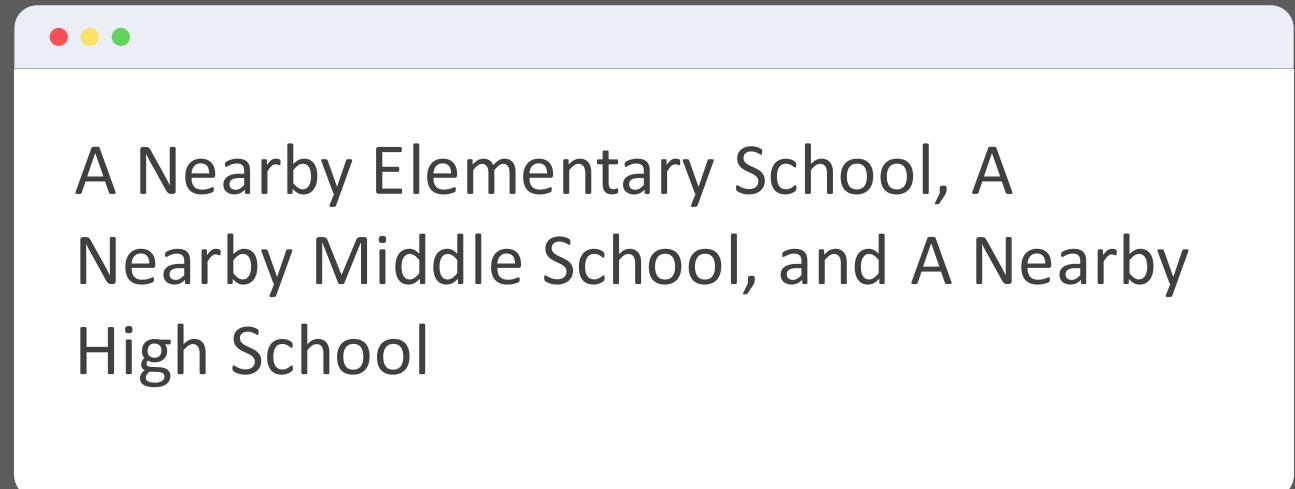
```
1  {% assign schoolList = 34 | FromCache:'DefinedType' %}  
2  {% assign schools = schoolList.DefinedValues %}  
3  {{ schools | Map:'Value' | Join:', ' | ReplaceLast:', ', and' }}
```





Shortening the Template

```
1  {% assign schools = 34 | FromCache:'DefinedType' | Property:'DefinedValues' %}  
2  {{ schools | Map:'Value' | Join:', ' | ReplaceLast:', ', and' }}
```





Shortening the Template

```
1 {{ 34 | FromCache:'DefinedType' | Property:'DefinedValues' | Map:'Value' | Join:', ' | ReplaceLast:', ',' and' }}
```



Code Smell

Is this understandable?
Will you need the records again?

A Nearby Elementary School, A
Nearby Middle School, and A Nearby
High School



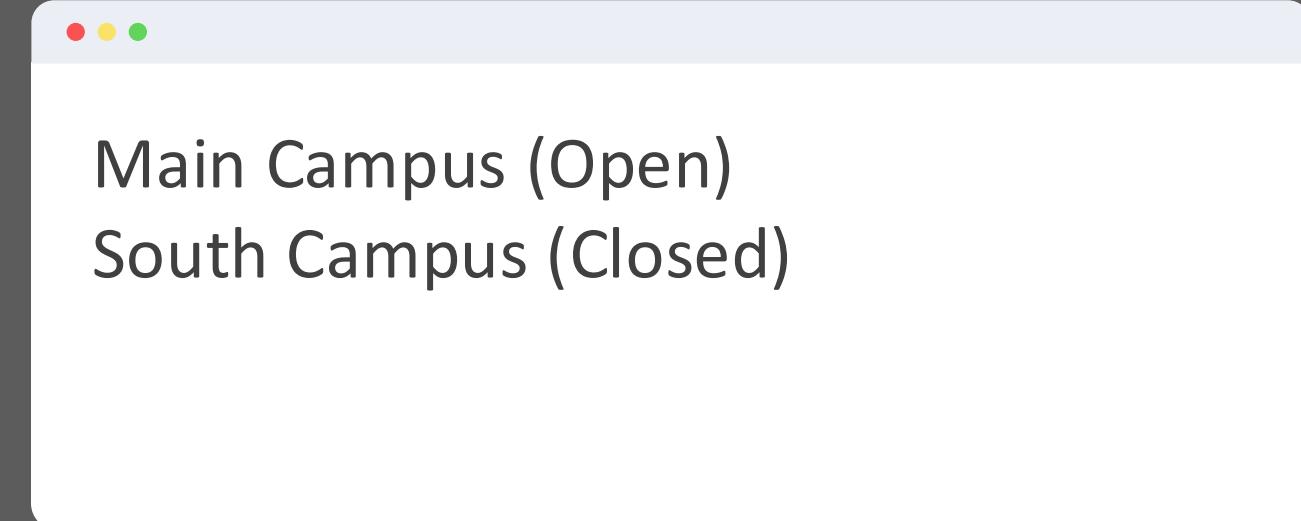
FromCache

```
1  {% assign campuses = 'All' | FromCache:'Campus' | Where:'CampusTypeId',768 %}
2  {% for campus in campuses %}
3    {{ campus.Name }} <small>({{ campus.CampusStatusValue.Value }})</small>
4  {% endif %}
```



Code Smell

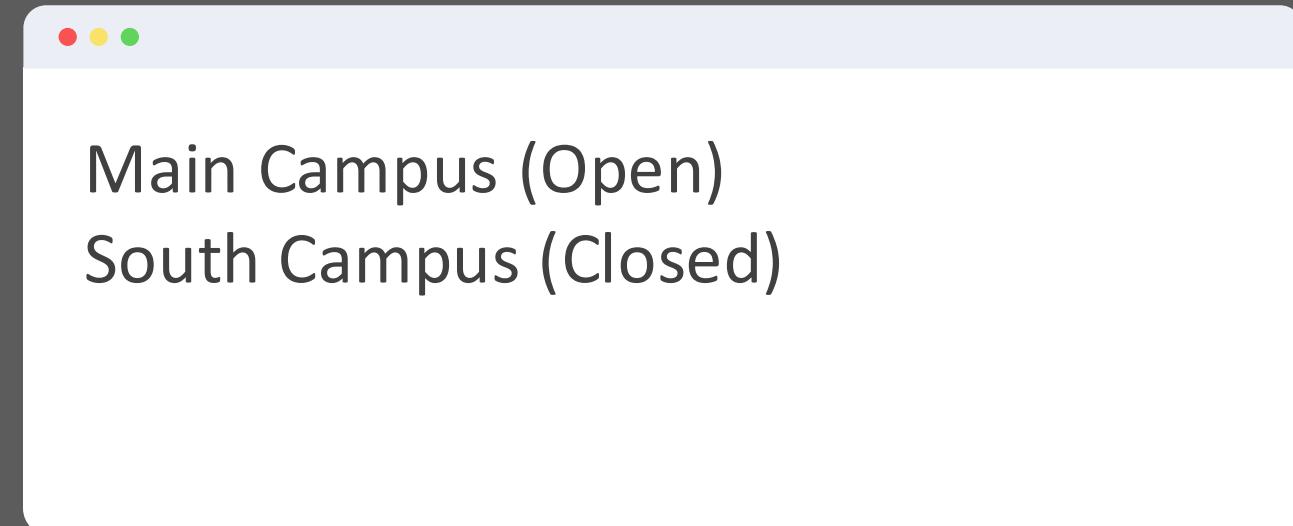
Are you going to understand
what 768 means next week?





FromCache

```
1  {% assign physicalCampusValueId = 768 %}  
2  //-----  
3  {% assign campuses = 'All' | FromCache:'Campus' | Where:'CampusTypeId',physicalCampusValueId %}  
4  {% for campus in campuses %}  
5    {{ campus.Name }} <small>({{ campus.CampusTypeValue.Value }})</small>  
6  {% endif %}
```





Adding to Cache Yourself

Most of the normal “content” blocks have cache settings.

- HTML block
- Content Channel View
- Content Channel Item View





Caching in Content Channel View Block

Channel Configuration

Channel i *

External Website Ads

Status i

 Pending Approval Approved Denied

Format i

```
1  {% include '~/Assets/Lava/AdList.lava' %}
```

Settings

Items Per Page i

3

Set Page Title i

Item Cache Duration i

1440

How long should Rock assume you haven't added or removed items, before it checks again?

Output Cache Duration i

60

How long should each of the items' content be held in cache before Rock checks for edits?

Cache Tags i

No cache tags defined.

Enable Archive Summary i



Caching in HTML Content Block

HTML Content CMS / Id: 347 x

Basic Settings **Advanced Settings**

Name • Footer Text

Enabled Lava Commands i

- All
- Adaptive Message
- Cache
- Calendar Events
- Event Scheduled Instance
- Execute
- Interaction Content Channel Item Write
- Interaction Intent Write
- Interaction Write
- Observe
- Rock Entity
- Search
- Sql
- Web Request
- Workflow Activate

Start in Code Editor mode i

Yes ▼

Document Root Folder i

~/Content ▼

Image Root Folder i

~/Content ▼

User Specific Folders i

No ▼

Cache Duration i

3600 ▼

How long should your template be held in cache before Rock checks for edits?



Partially-Cached Content



Personalization:

Be careful not to cache content that gets personalized, or else everyone using that page will see content for the first visitor.

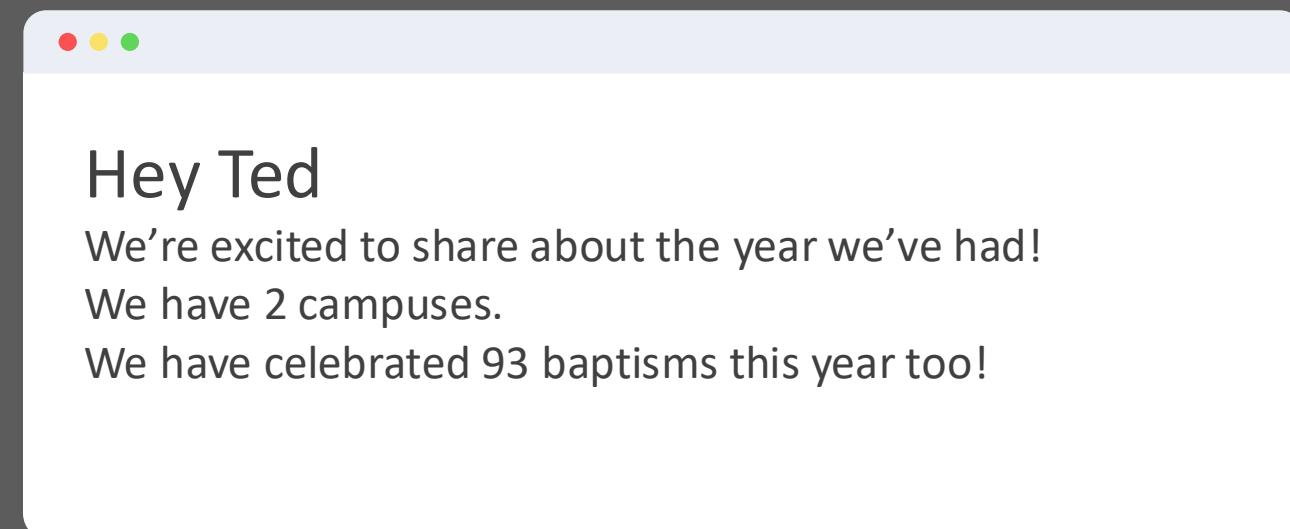
So ... how can I cache some, but not all, of a block's content?





Sample Template

```
1 <h3>Hey {{ CurrentPerson.NickName }}</h3>
2 <p>We're excited to share about the year we've had!</p>
3 <p>We have {{ Campuses | Size }} campuses.</p>
4 {% person where:'BaptismDate >= "2024-01-01"' count:'true' %}
5   <p>We have celebrated {{ count }} baptisms this year too!</p>
6 {% endperson %}
```



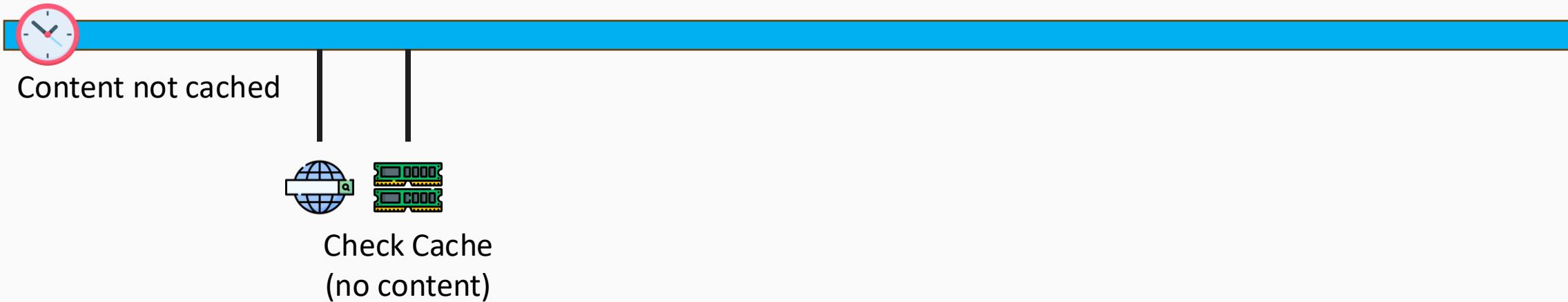


Caching



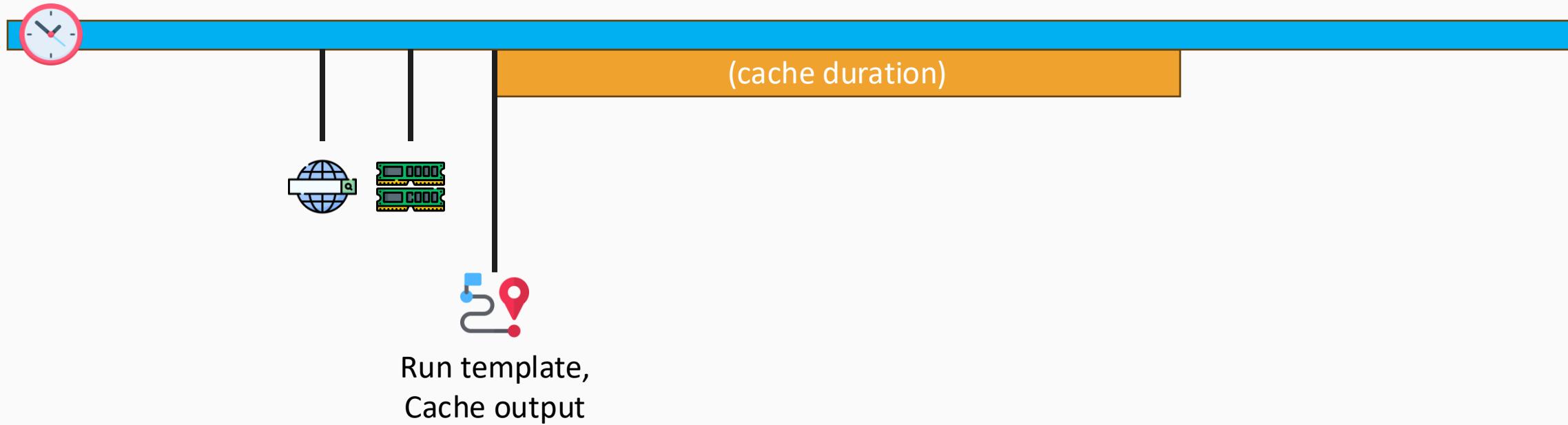


Caching



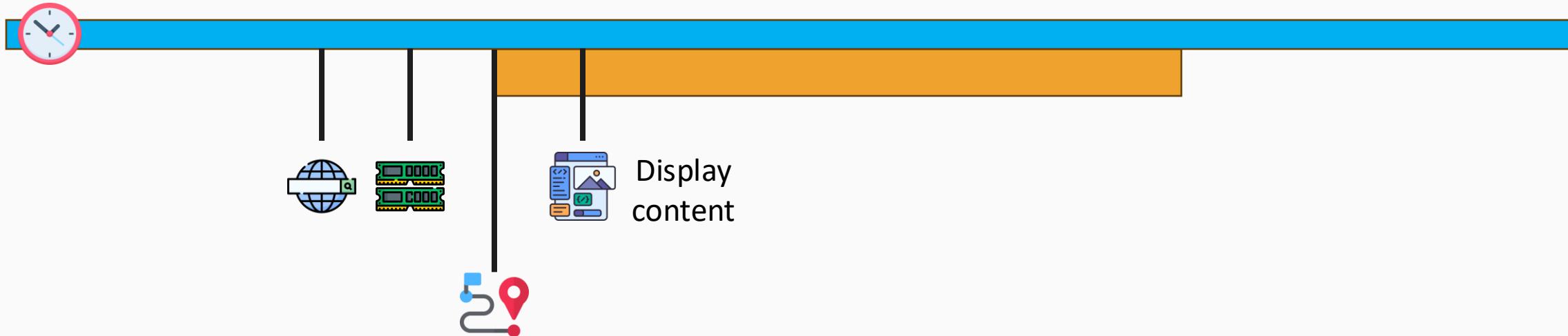


Caching



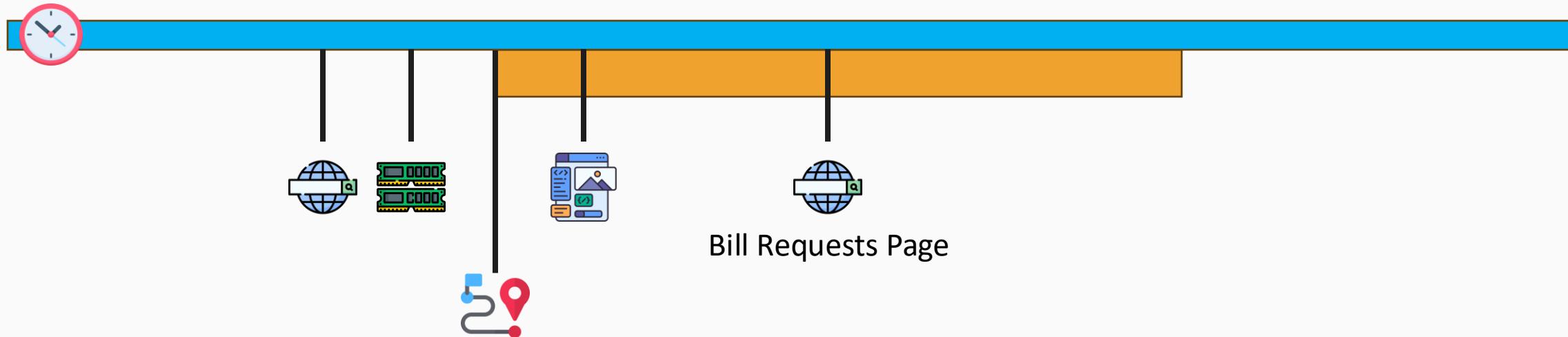


Caching



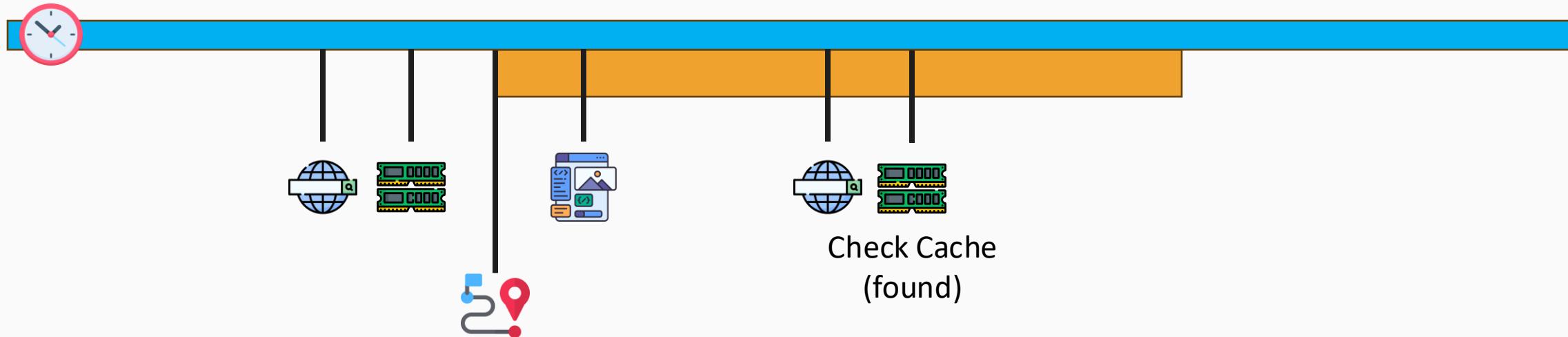


Caching



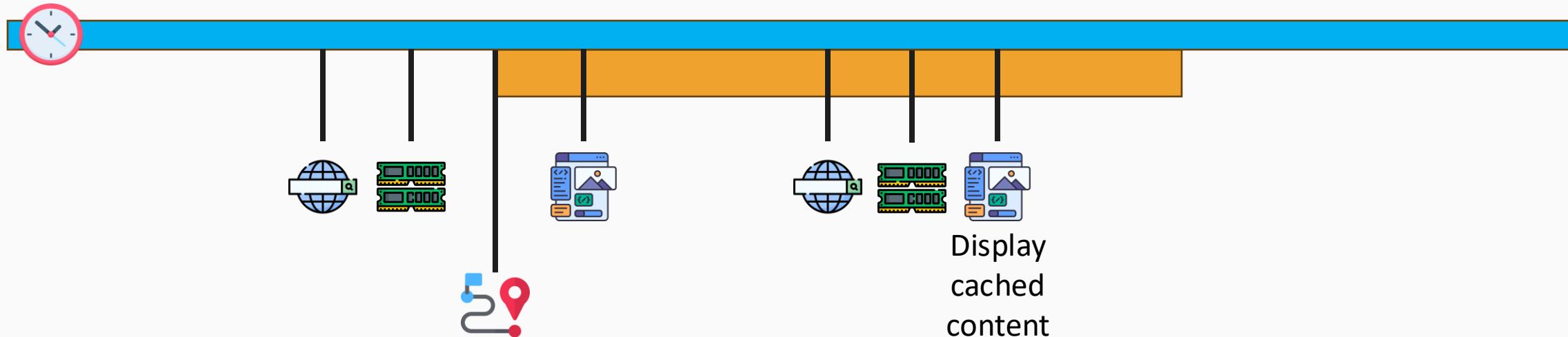


Caching



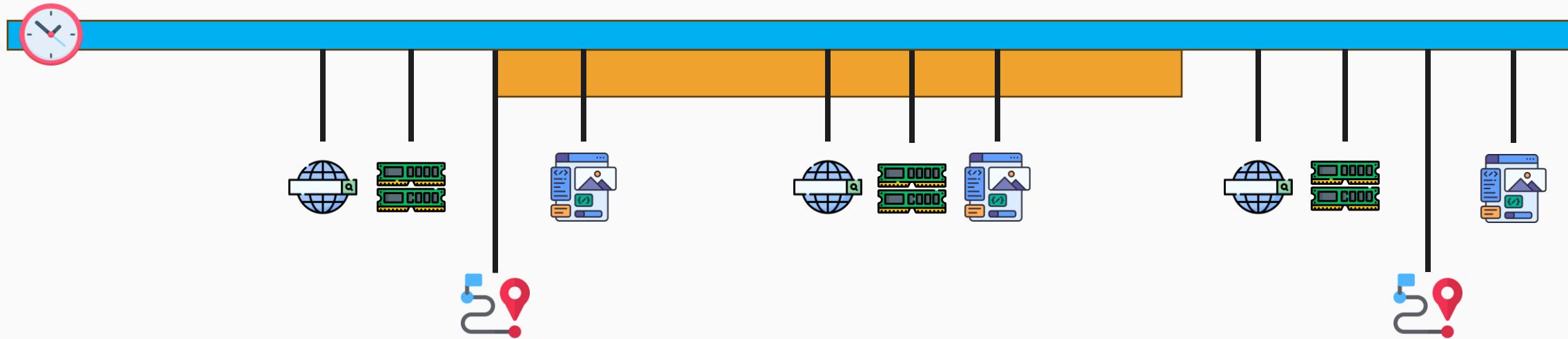


Caching





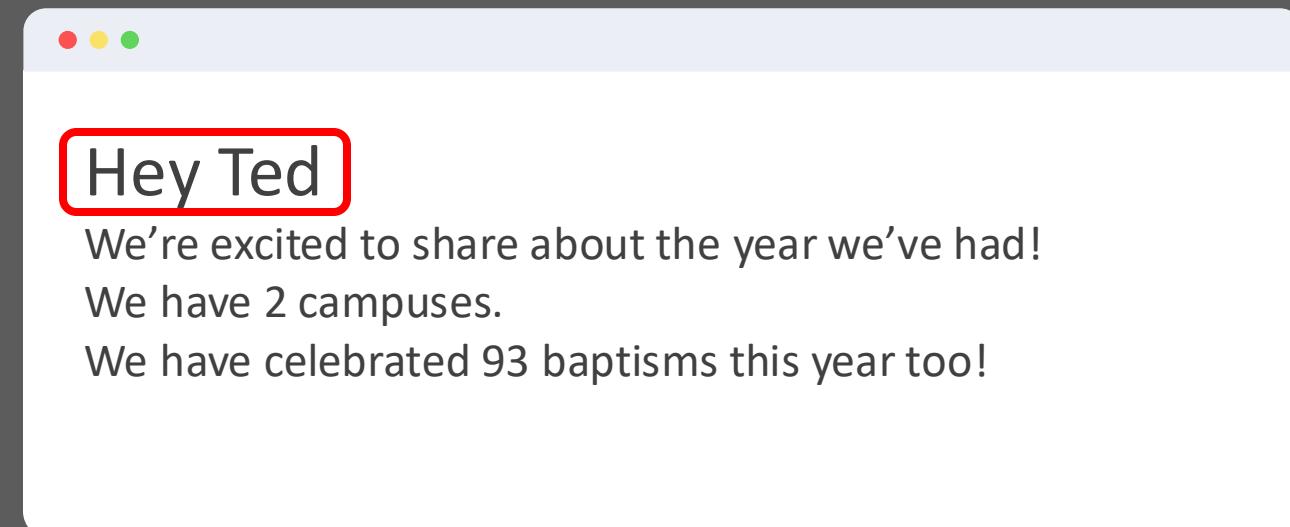
Caching





Sample Template

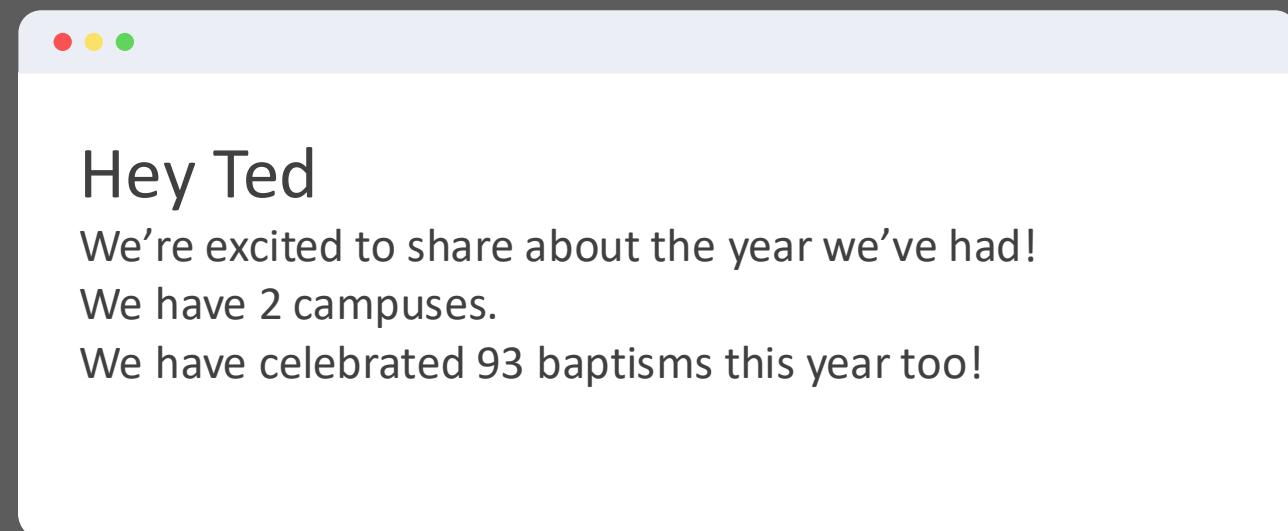
```
1 <h3>Hey {{ CurrentPerson.NickName }}</h3>
2 <p>We're excited to share about the year we've had!</p>
3 <p>We have {{ Campuses | Size }} campuses.</p>
4 {% person where:'BaptismDate >= "2024-01-01"' count:'true' %}
5   <p>We have celebrated {{ count }} baptisms this year too!</p>
6 {% endperson %}
```





Cache Tag

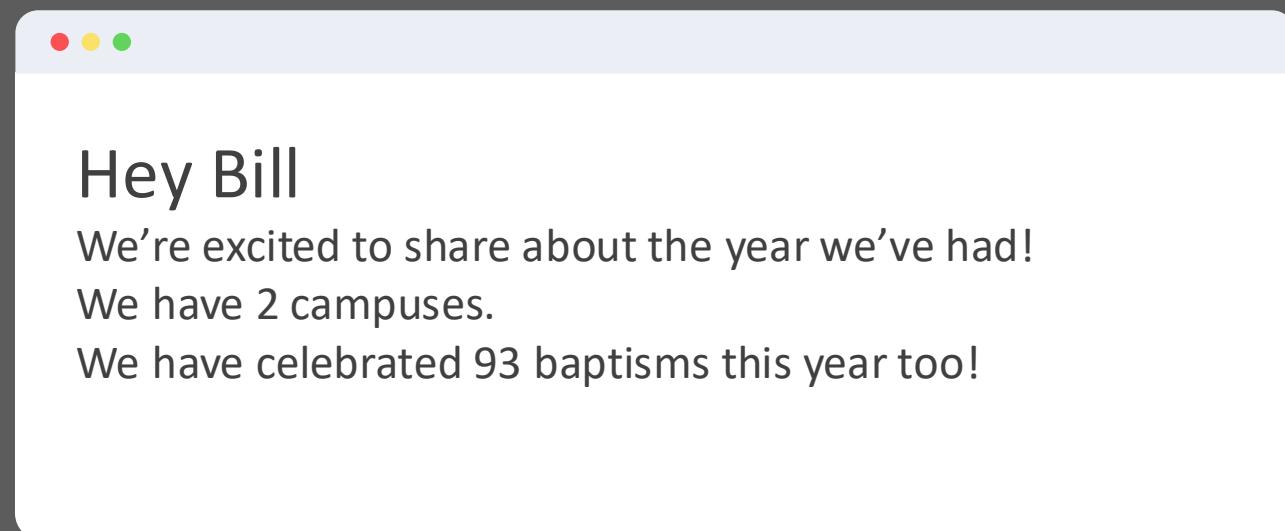
```
1 <h3>Hey {{ CurrentPerson.NickName }}</h3>
2 {% cache key:'yearStats' duration:'3600' %}
3   <p>We're excited to share about the year we've had!</p>
4   <p>We have {{ Campuses | Size }} campuses.</p>
5   {% person where:'BaptismDate >= "2024-01-01"' count:'true' %}
6     <p>We have celebrated {{ count }} baptisms this year too!</p>
7   {% endperson %}
8 {% endcache %}
```





Cache Tag

```
1 <h3>Hey {{ CurrentPerson.NickName }}</h3>
2 {% cache key:'yearStats' duration:'3600' %}
3   <p>We're excited to share about the year we've had!</p>
4   <p>We have {{ Campuses | Size }} campuses.</p>
5   {% person where:'BaptismDate >= "2024-01-01"' count:'true' %}
6     <p>We have celebrated {{ count }} baptisms this year too!</p>
7   {% endperson %}
8 {% endcache %}
```





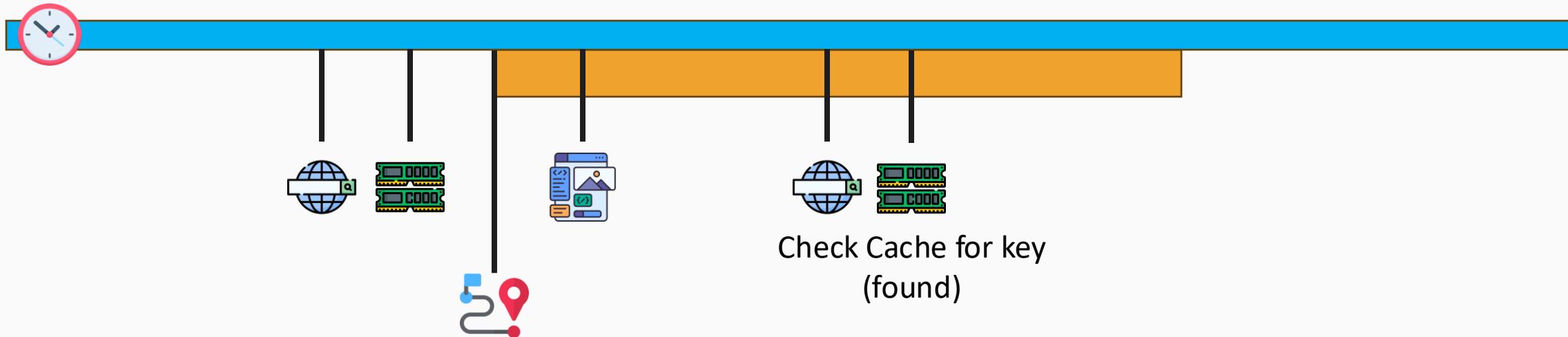
Cache Tag

```
1  {% cache key: '' duration: '' %}  
2  
3  {% endcache %}
```

- Instead of caching the contents by the block it's in, Rock will use the key you provide instead.



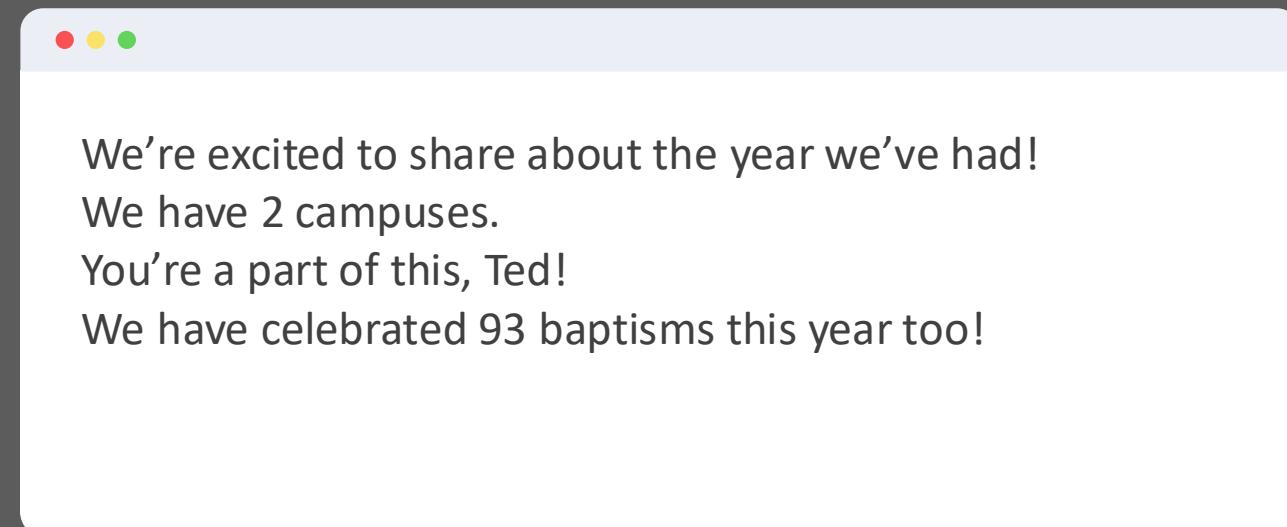
Caching





Cache Tag

```
1  {% cache key:'yearStats' duration:'3600' twopass:'true' %}  
2      <p>We're excited to share about the year we've had!</p>  
3      <p>We have {{ Campuses | Size }} campuses.</p>  
4      <p>You're a part of this, {{ raw }}{{ CurrentPerson.NickName }}{{ endraw }}!</p>  
5      {% person where:'BaptismDate >= "2024-01-01"' count:'true' %}  
6          <p>We have celebrated {{ count }} baptisms this year too!</p>  
7      {% endperson %}  
8  {% endcache %}
```





Raw Tag

```
1  {%- raw %}  
2  
3  {% endraw %}
```

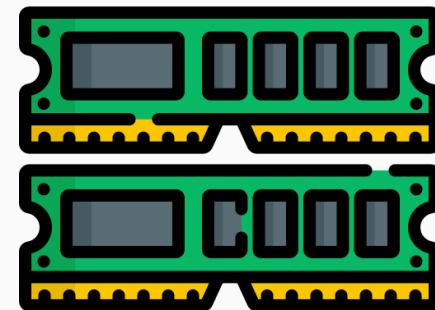
- The raw tag prevents Lava from getting interpreted – it outputs exactly what you typed.
- But naturally the raw tag doesn't get included.



Two Pass Caching

```
1  {% cache key:'yearStats' duration:'3600' twopass:'true' %}  
2    <p>We're excited to share about the year we've had!</p>  
3    <p>We have {{ Campuses | Size }} campuses.</p>  
4    <p>You're a part of this, {{ CurrentPerson.NickName }}!</p>  
5    {% person where:'BaptismDate >= "2024-01-01"' count:'true' %}  
6      <p>We have celebrated {{ count }} baptisms this year too!</p>  
7    {% endperson %}  
8  {% endcache %}
```

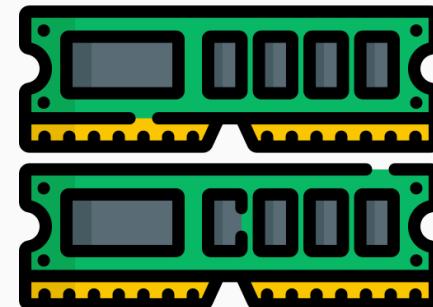
<p>We're excited to share about the year we've had!</p>
<p>We have 2 campuses.</p>
<p>You're a part of this, {{ CurrentPerson.NickName }}!</p>
<p>We have celebrated 93 baptisms this year too!</p>





Two Pass Caching

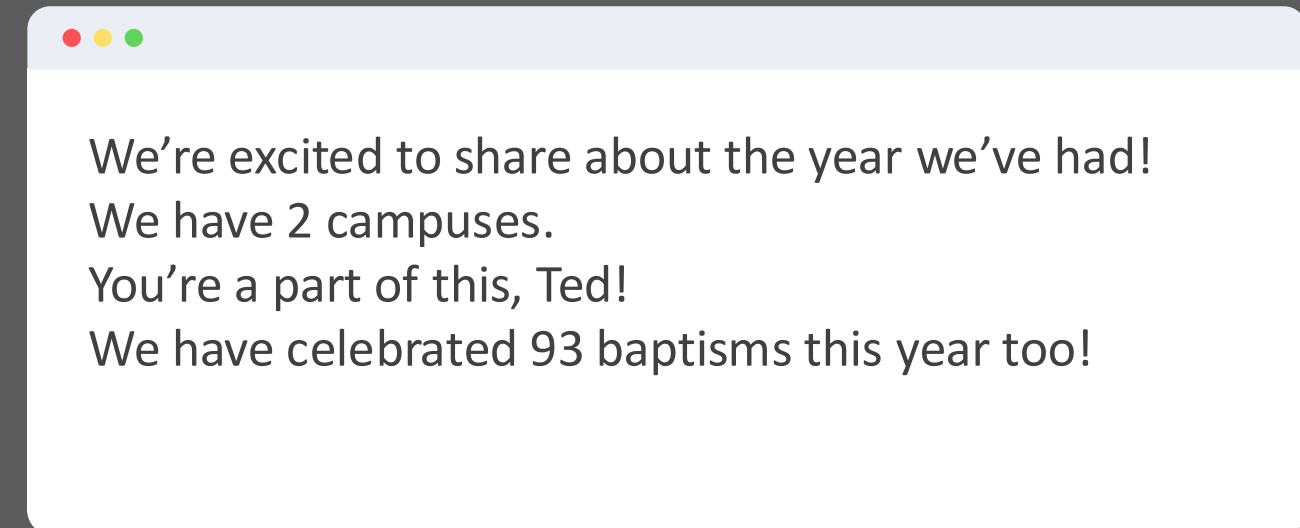
```
1  {% cache key:'yearStats' duration:'3600' twopass:'true' %}  
2      <p>We're excited to share about the year we've had!</p>  
3      <p>We have {{ Campuses | Size }} campuses.</p>  
4      <p>You're a part of this, {% raw %}{{ CurrentPerson.NickName }}{% endraw %}!</p>  
5      {% person where:'BaptismDate >= "2024-01-01"' count:'true' %}  
6          <p>We have celebrated {{ count }} baptisms this year too!</p>  
7      {% endperson %}  
8  {% endcache %}
```





Two Pass Caching

```
1 <p>We're excited to share about the year we've had!</p>
2 <p>We have 2 campuses.</p>
3 <p>You're a part of this, {{ CurrentPerson.NickName }}!</p>
4 <p>We have celebrated 93 baptisms this year too!</p>
```





Request Efficiency

Even when requests are cached,
you should try to make sure
they're as fast as possible.





Limit the Number of Database Lookups

- Remember the slowest request is a trip to the database.





Principles of Efficient Queries

```
1  {% for item in Items %}  
2      <h3>{{ item.Title }}</h3>  
3      <p>By {{ item | Attribute:'Author', 'FullName' }}</p>  
4  {% endfor %}
```



Attributes:

Every attribute lookup requires
a trip to the database

The screenshot shows a web browser window with two articles listed:

- This Year at Rock Solid Church**
By Pete Foster
- What's Happening in our Community?**
By Ted Decker



Principles of Efficient Queries

```
1  {% for person in People %}  
2    {% assign startingPointClasses = person | Steps:'1','Complete','2' %}  
3    {{ person.NickName }} has taken Starting Point with:  
4    <ul>  
5      {% for class in startingPointClass %}  
6        <li>  
7          {{ class | Attribute:'Leader','FullName' }}  
8          ({{ class.CompletedDateTime }})  
9        </li>  
10      {% endfor %}  
11    </ul>  
12  {% endfor %}
```

Requires a lookup for every person in the loop

Requires a lookup for every class each person has taken

The screenshot shows a Mac OS X application window with three colored window control buttons (red, yellow, green) at the top. Inside the window, there are two sections of text:

Ted has taken Starting Point with:

- Pete Foster (6/7/2003)
- Daniel Peak (2/29/2012)

Sarah has taken Starting Point with:

- Pete Foster (11/3/2019)



Principles of Efficient Queries

```
1  {% for person in People %}  
2      
3    {{ person.NickName }}  
4  {% endfor %}
```

Start with the person record

- └ Request the Family
 - └ Request the Campus
 - └ Request the Location
 - └ Request the Image -> to get the URL property.



Let Multiple Dots be a Mental Flag

Frequently when there is more than one “dot”, it means you’re going from entity to entity. Each entity lookup is a separate trip to the database.



Are Templates like this “Bad”?

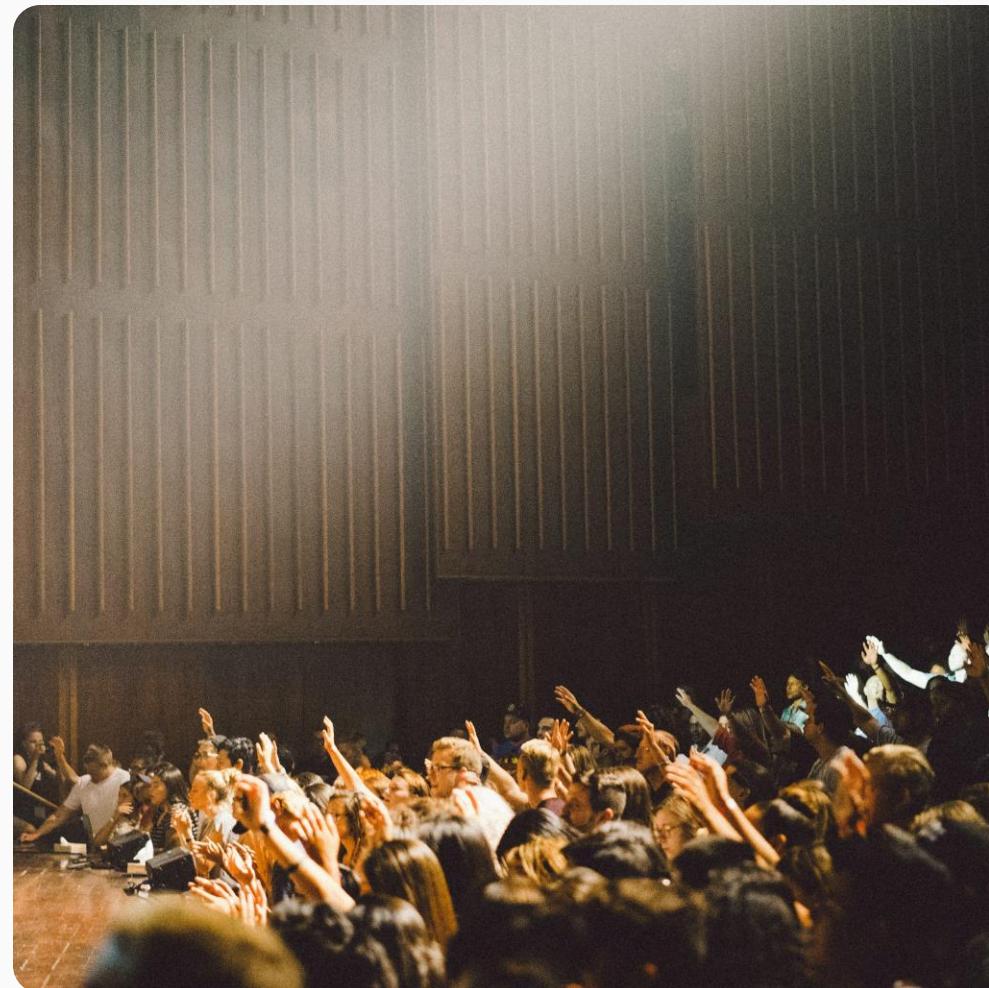
- It's all about context.
- Be aware of what we're asking the server to do.
- The scale of a page's use makes a huge difference!





Principles of scaling

- Things work great when there's just one person using the page.
- What changes when 1,000 people load it at the same time?
- What happens when the cache expires?





Persisted Dataset

A tool for proactively getting and caching the facts you need.

- Stored as JSON
- Refreshed on a schedule, not on load.

