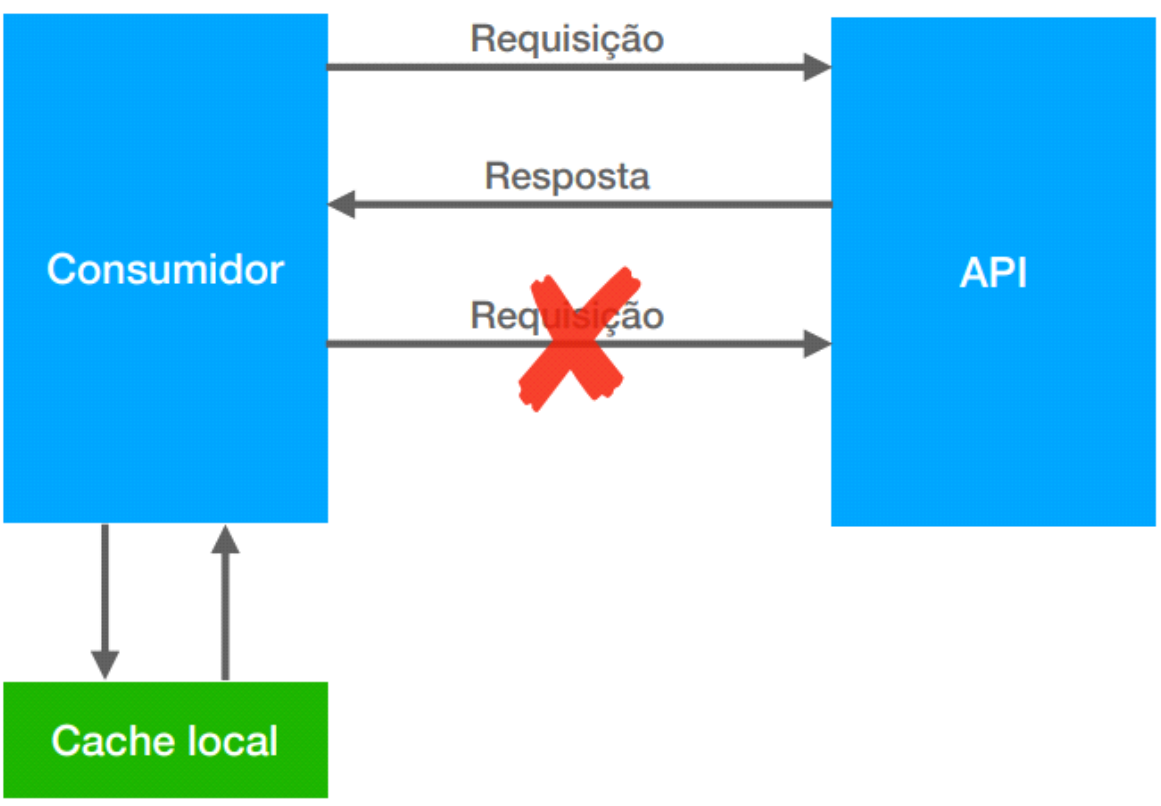
17.1. Introdução ao Cache de HTTP

sábado, 8 de abril de 2023 10:26

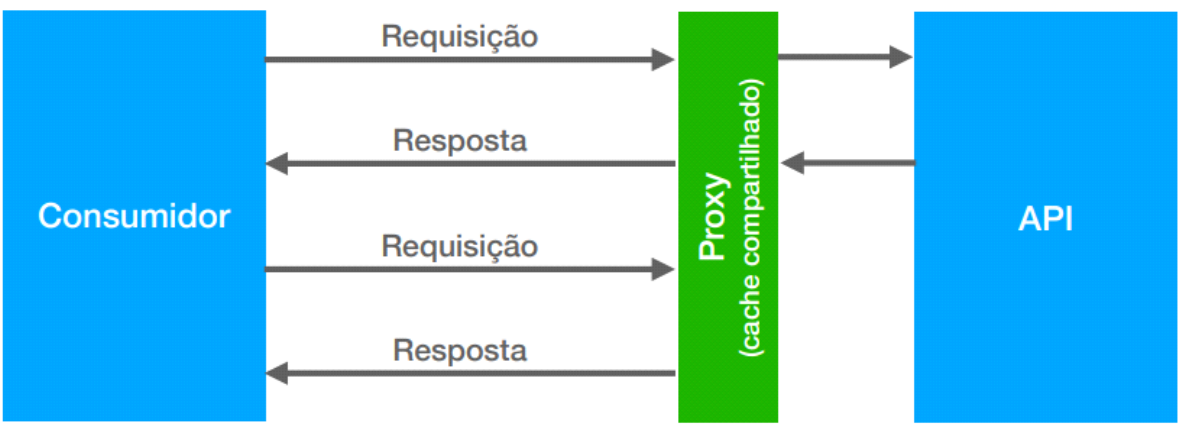
- Otimizar tempo de resposta
- Aproveitar a infraestrutura de http
- Habilidade que componentes de softwares tem de armazenar dados que são usados frequentemente
- Cache HTTP
- Especificação HTTP para servidores implementar

Cache Local:



- Em referências, a API pode ser chamado de Origin Server
- O servidor estipula o tempo para que o recurso possa se manter em cache local
- Para recursos ainda no limite de tempo do cache, são chamados cache frescos e recursos antigos no cache, stail cache

Cache Compartilhados



- Caches públicos/compartilhados servem a mais de um cliente
- O cliente não deixa de ter o cache local
- O tempo também é estipulado pela API

Benefícios

- Reduz uso de banda
- Reduz latência
- Reduz carga no servidor
- Esconde problemas na rede

Quando não fazer cache

- Consumidores não toleram diferenças de estados dos recursos do cache e origin server

17.2. Habilitando o cache com o cabeçalho Cache-Control e a diretiva max-age

sábado, 8 de abril de 2023 11:19

```java
@GetMapping
public ResponseEntity<List<FormaPagamentoDTO>> listar(){
    final List<FormaPagamentoDTO> formaPagamentosDTO = fPAssembler.toListDTO(f
    return ResponseEntity.ok()
            .cacheControl(CacheControl.maxAge( maxAge: 10, TimeUnit.SECONDS))
            .body(formaPagamentosDTO);
}
```

Em um retorno de ResponseEntity, há o método cacheControl que recebe uma instância de CacheControl.

O CacheControl usa o builder, então podemos chamar diretamente os métodos e passar os argumentos, no caso maxAge para especificar a quantidade de tempo, e um TimeUnit para unidade de tempo

Name	Status	Type	Initiator	Size	Time
formapagamentos	200	xhr	jquery-3.4....	439 B	33 ms
formapagamentos	200	xhr	jquery-3.4....	(disk cache)	3 ms
formapagamentos	200	xhr	jquery-3.4....	(disk cache)	22 ms
formapagamentos	200	xhr	jquery-3.4....	439 B	20 ms
formapagamentos	200	xhr	jquery-3.4....	(disk cache)	7 ms

Nota: o Postman não oferece cache local, uma alternativa é usar Talend API Tester - Free Edition como extensão do Google Chrome

17.3. Desafio: adicionando o cabeçalho Cache-Control na resposta

sábado, 8 de abril de 2023 11:35

```java
@GetMapping("/{formaPagamentoId}")
public ResponseEntity<FormaPagamentoDTO> buscar(@PathVariable Long formaPagamentoId){
    final FormaPagamento formaPagamento = formaPagamentoService.buscar(formaPagamentoId);
    final FormaPagamentoDTO formaPagamentoDTO = fPAssembler.toDTO(formaPagamento);
    return ResponseEntity.ok()
            .cacheControl(CacheControl.maxAge( maxAge: 10, TimeUnit.SECONDS))
            .body(formaPagamentoDTO);
}
```

17.4. Entendendo a validação de representações em cache com Etags

sábado, 8 de abril de 2023 11:39

- Entity Tag
- Reaproveitar a resposta cache no estado stale

O HTTP provê no cabeçalho (header) da resposta (response), a ETag que é um identificador para uma versão específica de um recurso. A ETag permite que o cache torne-se mais eficiente e preserve o tráfego de dados (largura de banda), assim um web server não precisa reenviar uma resposta com todos os dados que não tiveram nenhuma mudança em seu conteúdo. Além disso, as ETags ajudam a impedir que atualizações simultâneas de um recurso sejam feitas por outros.

Se o recurso numa URL sofre mudança, a Etag assume um novo valor que deve ser gerado pelo Web Server. Uma comparação entre elas podem determinar se as duas representações do recurso são iguais. Etags são similares às nossas impressões digitais, e por isso também podem ser usadas por alguns servidores como um forma de rastreamento. Elas podem ser configuradas a fim de que possam ser persistidas idenfinidamente por um servidor de rastreamento.

A requisição é enviada ao servidor com a Etag da representação do recurso em estado stale, ou seja, a etag com o último recurso disponível em estado stale, a requisição condicional leva a etag e o servidor verifica se o conteúdo da etag é a mesma da nova representação, caso verdadeiro, a resposta não vem com o payload e com o código 304 Not Modified indicando que não há necessidade de retransmitir a requisição de recursos. Caso falso, a nova requisição é validada com falso, e envia a nova representação atual no payload da resposta.

17.5. Implementando requisições condicionais com Shallow Etags

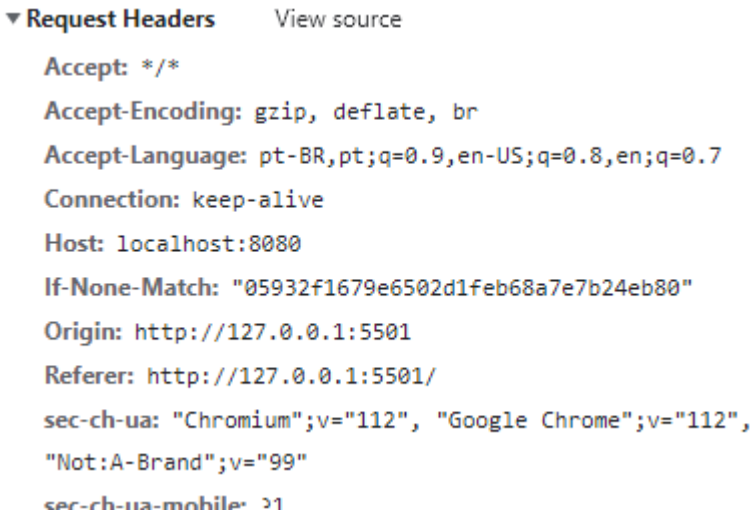
sábado, 8 de abril de 2023 14:45

```java
package com.algaworks.algafood.core.web;

import ...

@Configuration
public class WebConfig implements WebMvcConfigurer {

    @Override
    public void addCorsMappings(CorsRegistry registry) {...}

    @Bean
    public Filter shallowEtagHeaderFilter(){
        return new ShallowEtagHeaderFilter();
    }
}
```

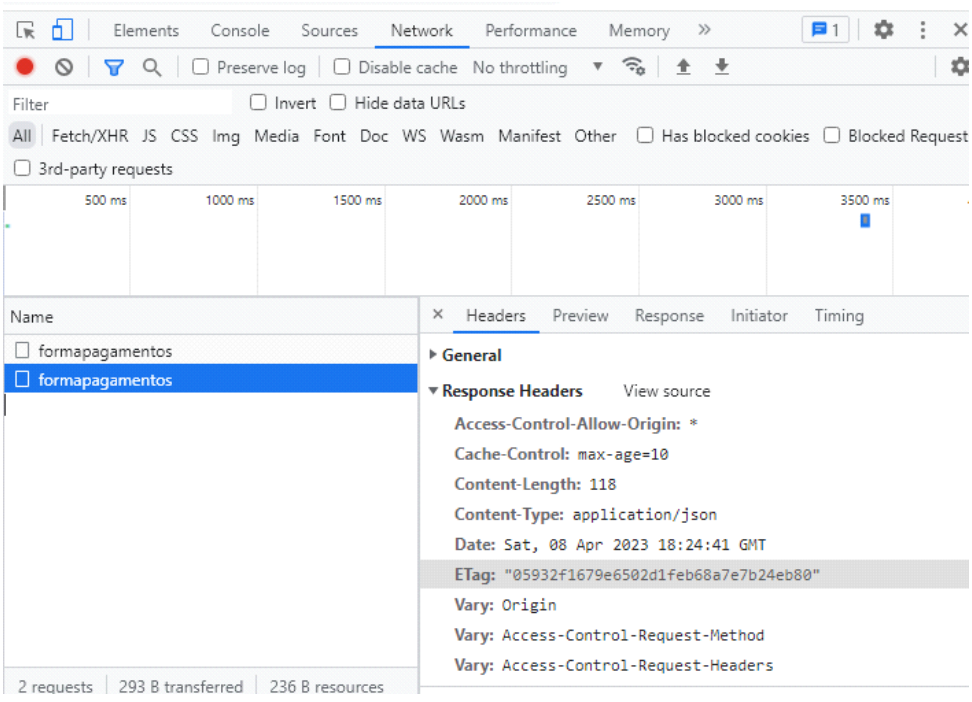
Filter: interface javax.servlet
ShallowEtagHeaderFilter: org.springframework.web.filter.ShallowEtagHeaderFilter

O filtro intercepta as requisições recebidas e respostas enviadas do server e diciona nos cabeçalhos da respostas a propriedade Etag com hash e valida se os hashs coincidem

1º requisição: servidor com banco de dados
a requisição terá no header If-None-Match caso a requisição anterior conter uma Etag no seu Header



2º requisição: cache local com eTag



A diferença é o envio do payload. Caso o hash da Etag serem iguais, uma resposta é enviada com o status 304, mas muitos navegadores omitem para o 200 ok, ou a resposta esperada, a' diferença é que não é enviado um corpo a menos que os hashs não sejam iguais.

17.6. Adicionando outras diretivas de Cache-Control na resposta HTTP

sábado, 8 de abril de 2023 20:45

- cachePrivate
  - Somente cache local do browser
- cachePublic
  - Cache local e cache público/servidor/proxy de cache público
- noCache
  - Sempre fazer validação do Etag, como se o cache estivesse sempre no estado stale
- noStore
  - Resposta não cacheavel. Nenhuma ferramenta pode armazenar o cache da resposta

```java
@GetMapping
public ResponseEntity<List<FormaPagamentoDTO>> listar(){
    final List<FormaPagamentoDTO> formaPagamentosDTO = fPAssembler.toListDTO(formaPagamentoService.listar());
    return ResponseEntity.ok()
            .cacheControl(CacheControl.maxAge(10, TimeUnit.SECONDS).cachePrivate())
            .cacheControl(CacheControl.maxAge( maxAge: 10, TimeUnit.SECONDS).cachePublic())
            .cacheControl(CacheControl.noCache())
            .cacheControl(CacheControl.noStore())
            .body(formaPagamentosDTO);
}
```

17.7. Usando a diretiva no-cache no cabeçalho Cache-Control da requisição

sábado, 8 de abril de 2023 21:23

Para forçar a requisição no origin server, basta apenas adicionar no cabeçalho da requisição (no cliente) a propriedade cache-control com o valor no-cache.

17.8. Entendendo e preparando a implementação de Deep Etags

sábado, 8 de abril de 2023 21:29

- Evita que todo o processamento seja feito no lado do servidor diante da implementação do filtro Shallow Etag.
- Diferente do uso do filtro do Shallow Etag (17.5. Implementando requisições condicionais com Shallow Etags) o Deep Etags também retorna um 304 not modified como não faz processamento do servidor e do banco de dados para realizar a consulta caso as Etags sejam idênticas
- Além de economizar banda, economiza processamento do servidor e banco de dados

Ideia de iImplementação:

- Consulta de Etag na entidade FormPagamento
- Gerar hash de um atributo na entidade (data_atualização) e não de o recurso da requisição
- Caso o cache local estiver stale, verifica com Etag o estado do recurso do servidor através de uma Etag própria, desviando a possibilidade de buscar todo o recurso no banco de dados e fazer a verificação do recurso
- Adicionar um atributo de tempo na entidade FormaPagamento afim de verificar a última atualização e fazer a busca da última atualização de forma pagamento no banco de dados, caso os horários de atualização sejam os mesmos, retornar uma resposta 304 not modified, caso contrário, fazer a busca do recurso atualizado.

- Migration V012

```
Script
1  alter table forma_pagamento
2      add column data_atualizacao datetime null;
3  update forma_pagamento
4  set forma_pagamento.data_atualizacao = utc_timestamp;
5  alter table forma_pagamento
6      modify column data_atualizacao datetime not null;
```

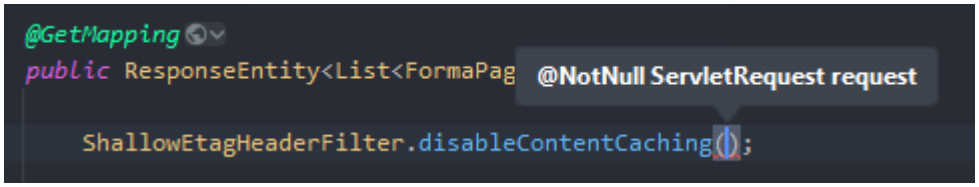
- AfterMigrate

```
54  insert into forma_pagamento, data_atualizacao (id, descricao, utc_timestamp)
55  values (1, 'Cartão de crédito');
56  insert into forma_pagamento, data_atualizacao  (id, descricao, utc_timestamp)
57  values (2, 'Cartão de débito');
58  insert into forma_pagamento, data_atualizacao  (id, descricao, utc_timestamp)
59  values (3, 'Dinheiro');
```

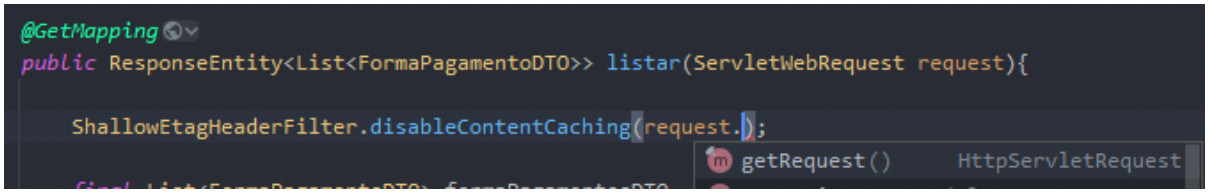
17.9. Implementando requisições condicionais com Deep Etags

domingo, 9 de abril de 2023 09:48

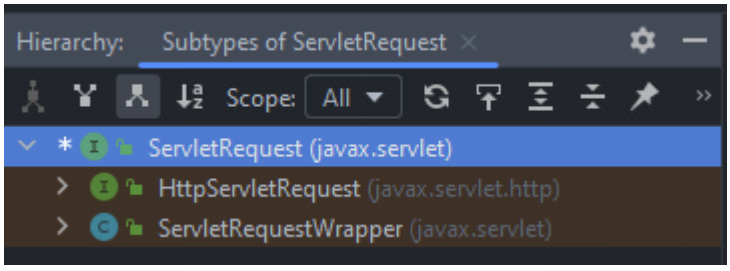
- Para implementar Deep Etags é necessário desabilitar Shallow Etags
- No método controller de forma de pagamento desabilitar Shallow Etags



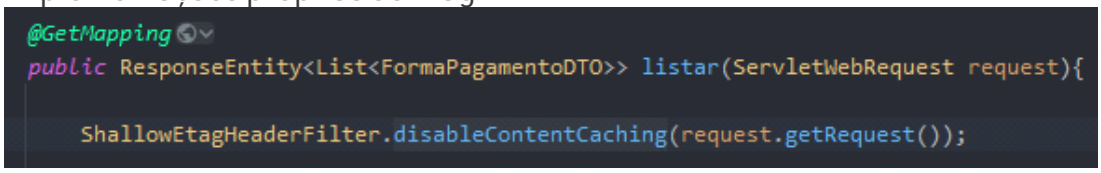
  - O método necessita de uma instância de ServletRequest como argumento
  - Podemos injetar no método controlador de listagem de forma de pagamento um ServletWebRequest



  - o objeto request tem um método para fazer um get na ServletWebRequest que retorna um HttpServletRequest, que é um ServletRequest



- Caso não haja implementação de desabilitação, o Shallow Etag substitui as implementações próprias de Etag



- Criar JQPL para retornar o último valor de um registro atualizado

```java
@Repository
public interface FormaPagamentoRepository extends JpaRepository<FormaPagamento, Long> {

    1 usage
    @Query("select max(dataUltimaAtualizacao) from FormaPagamento")
    OffsetDateTime getDataUltimaAtualizacao();
}
```

- A JPQL é usada para obter a data/hora mais recente na coluna "dataUltimaAtualizacao" da tabela "FormaPagamento". Essa query retorna o valor máximo da coluna "dataUltimaAtualizacao", ou seja, a data/hora mais recente presente na tabela. Essa data/hora pode ser usada para determinar quando a tabela foi atualizada pela última vez.
  Caso não haja registros na tabela, a query irá retornar um valor nulo.

```java
@GetMapping
public ResponseEntity<List<FormaPagamentoDTO>> listar(ServletWebRequest request){
    ShallowEtagHeaderFilter.disableContentCaching(request.getRequest());

    String eTag = "0";

    final OffsetDateTime dataUltimaAtualizacao = formaPagamentoRepository.getDataUltimaAtualizacao();

    if(dataUltimaAtualizacao != null){
        eTag = String.valueOf(dataUltimaAtualizacao.toEpochSecond());
    }

    final List<FormaPagamentoDTO> formaPagamentosDTO = fPAssembler.toListDTO(formaPagamentoService.li
    return ResponseEntity.ok()
            .cacheControl(CacheControl.maxAge(10, TimeUnit.SECONDS).cachePrivate())
           .cacheControl(CacheControl.maxAge( maxAge: 10, TimeUnit.SECONDS).cachePublic())
            .cacheControl(CacheControl.noCache())
            .cacheControl(CacheControl.noStore())
           .body(formaPagamentosDTO);
}
```

Caso a "dataUltimaAtualizacao" seja nula, quer dizer que não há registros e o padrão da variável eTag fica "0".

a ideia foi transformar com .toEpochSecond esta data-hora no número de segundos a partir da época de 1970-01-01T00:00:00Z. Passando a Etag no cabeçalho da resposta.

```java
return ResponseEntity.ok()
            .cacheControl(CacheControl.maxAge(10, TimeUnit.SECONDS).cachePrivate())
           .cacheControl(CacheControl.maxAge( maxAge: 10, TimeUnit.SECONDS).cachePublic())
            .cacheControl(CacheControl.noCache())
            .cacheControl(CacheControl.noStore())
           .header(HttpHeaders.ETAG, eTag)
           .body(formaPagamentosDTO);
```

ou

```
return ResponseEntity.ok()
          .cacheControl(CacheControl.maxAge(10, TimeUnit.SECONDS).cachePrivate())
        .cacheControl(CacheControl.maxAge( maxAge: 10, TimeUnit.SECONDS).cachePublic())
          .cacheControl(CacheControl.noCache())
          .cacheControl(CacheControl.noStore())
        .eTag(eTag)
        .body(formaPagamentosDTO);
```

Ficaremos com a última alternativa.

Caso não haja intervenção explicita do ShallowEtagHeaderFilter, o mesmo adiciona na resposta um header com o valor automático do etag.

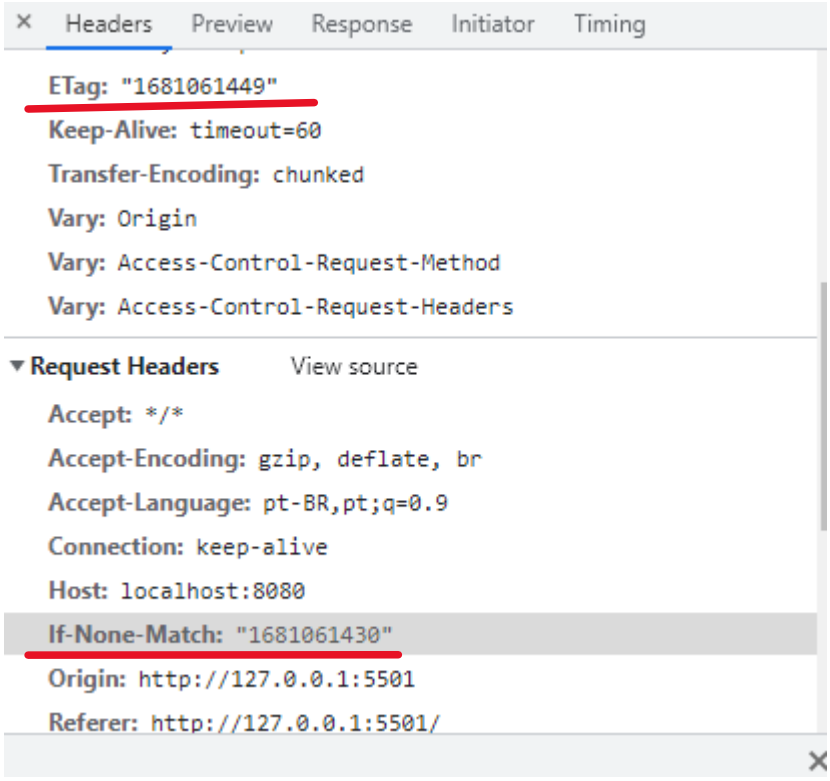
Mas agora estamos adicionando manualmente um valor eTag no header eTag.

Já

Agora precisamos fazer a verificação da requisição com a etag no header If-None-Match, como foi visto na aula [17.4. Entendendo a validação de representações em cache com Etags]

e se o valor do If-None-Match coincidir com o valor da eTag (dataAtualização), retornamos null, e o Spring trata a resposta para retornar com o status 304 not modified

```java
@GetMapping
public ResponseEntity<List<FormaPagamentoDTO>> listar(ServletWebRequest request){
    ShallowEtagHeaderFilter.disableContentCaching(request.getRequest());

    String eTag = "0";

    final OffsetDateTime dataUltimaAtualizacao = formaPagamentoRepository.getDataUltimaAtualizacao();

    if(dataUltimaAtualizacao != null){/*Pode vir nulo caso não haja forma de pagamento*/
        eTag = String.valueOf(dataUltimaAtualizacao.toEpochSecond());
    }

    if(request.checkNotModified(eTag)){/*true para etag iguais*/
        return null;
    }
```



17.10. Desafio: implementando requisições condicionais com Deep Etags

domingo, 9 de abril de 2023 13:54

- Adicionar no parâmetro da requisição no controlador, um objeto tipo ServletWebRequest

```
@GetMapping("/{formaPagamentoId}")
public ResponseEntity<FormaPagamentoDTO> buscar(@PathVariable Long formaPagamentoId, ServletWebRequest request){
    final FormaPagamento formaPagamento = formaPagamentoService.buscar(formaPagamentoId);
    ShallowEtagHeaderFilter.disableContentCaching(request.getRequest());
```

- Implementar no repositório uma query para buscar o atributo que servirá como etag
  - data mais recente da última atualização de um atributo no banco de dados

```
@Repository
public interface FormaPagamentoRepository extends JpaRepository<FormaPagamento, Long> {

    1 usage
    @Query("select max(dataAtualizacao) from FormaPagamento")
    OffsetDateTime getDataUltimaAtualizacao();

    1 usage
    @Query("select fP.dataAtualizacao from FormaPagamento fP where fP.id =:formaPagamentoId")
    OffsetDateTime getDataUltimaAtualizacaoId(Long formaPagamentoId);
}
```

- Desabilitar ShallowHeaderFilter chamando o método disableContentCaching e no argumento um objeto ServletRequest vindo de um objeto tipo ServletWebRequest chamando o método getRequest

```
ShallowEtagHeaderFilter.disableContentCaching(request.getRequest());
```

- Verificar se as eTags da requisição e do banco de dados são iguais com o método ServletWebRequest.checkNotModified

```
final OffsetDateTime dataUltimaAtualizacaoId = formaPagamentoRepository.getDataUltimaAtualizacaoId(formaPagamentoId);
if(dataUltimaAtualizacaoId != null)
    eTag = String.valueOf(dataUltimaAtualizacaoId.toEpochSecond());

if(request.checkNotModified(eTag))
    return null;
```

- Caso não sejam iguais, retornar no header um eTag novo com o recurso modificado

```
final FormaPagamentoDTO formaPagamentoDTO = fPAssembler.toDTO(formaPagamento);
return ResponseEntity.ok()
        .cacheControl(CacheControl.maxAge( maxAge: 10, TimeUnit.SECONDS))
        .eTag(eTag)
        .body(formaPagamentoDTO);
```