Tarefas agendadas

Para usar **métodos agendados** em uma aplicação Spring, primeiro é necessário **fazer a habilitação anotando a classe main** do projeto:

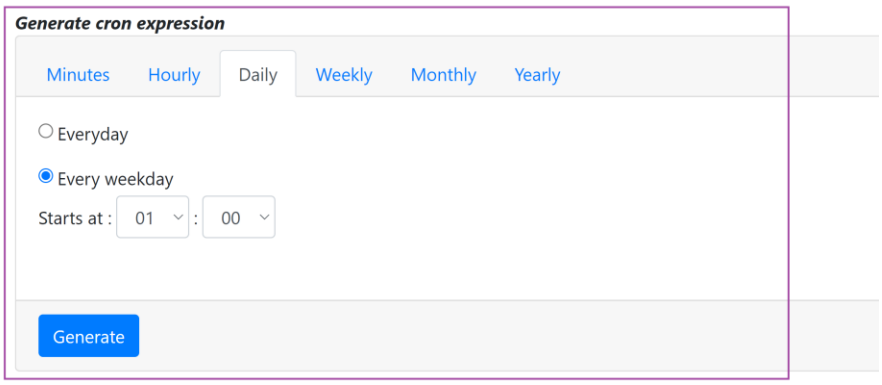
```java
@EnableSpringDataWebSupport
@SpringBootApplication
@EnableScheduling
public class TodoApplication {

    public static void main(String[] args) { SpringApplication.run(TodoApplication.class, args); }
```

A anotação **@Scheduled** vai indicar pro Spring que esse método anotado é um método agendado:

```java
@Scheduled(cron = "")
public void enviarEmailMetaNaoConcluidaDiaFinal(){

}
```

Precisamos passar a "frequência" com que ele será executado, fazemos isso por parâmetro usando a propriedade **"cron"**. Essa propriedade **"cron" é um valor cronológico** do tipo String que indica a ***hora, minuto, dia, mês, ano...*** em que esse método vai ser executado.

Basicamente essa propriedade recebe o **segundo, minuto, hor, dias, mês e ano = "0 0 0 1 1 1".** Mas tem um site que gera esse tipo de expressão de uma maneira mais clara, esse site é o **cronMaker:**



Com a **expressão gerada** pelo **CronMaker** eu posso atribuir a uma variável e usar como parâmetro do meu método agendado:

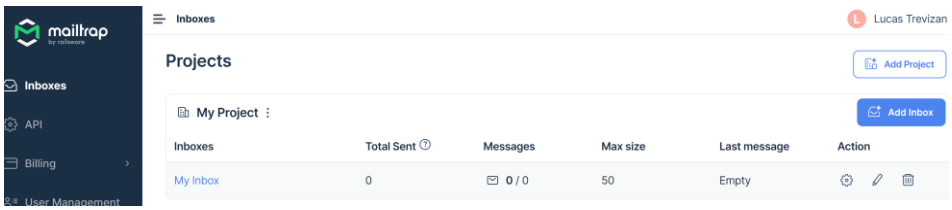
```java
private static final String CRON_META_PRAZO_FINAL = "0 0 1 ? * MON-FRI";

@Scheduled(cron = CRON_META_PRAZO_FINAL)
public void enviarEmailMetaNaoConcluidaDiaFinal(){

}
```

Dessa maneira eu **configuro um método que será executado todo dia as 1 da manhã**, agora basta **implementar um corpo para esse método**. No caso eu vou buscar as metas que vencem no dia e enviar uma mensagem para o email de cada usuário os alertando:

```java
@Scheduled(cron = CRON_META_PRAZO_FINAL)
public void enviarEmailMetaNaoConcluidaDiaFinal(){

    LocalDateTime inicioDia = hoje.toLocalDate().atStartOfDay();
    LocalDateTime fimDia = hoje.with(LocalTime.of( hour: 23,  minute: 59,  second: 59));

    List<Meta> metasNoPrazoFinal = metaRepository.listarMetasNoDiaFinal(inicioDia, fimDia);

    List<String> emails = metasNoPrazoFinal
            .stream()  Stream<Meta>
            .map(meta -> meta.getUsuario().getEmail())  Stream<String>
            .collect(Collectors.toList());

    emailService.enviarEmails(mensagem, emails);
}
```

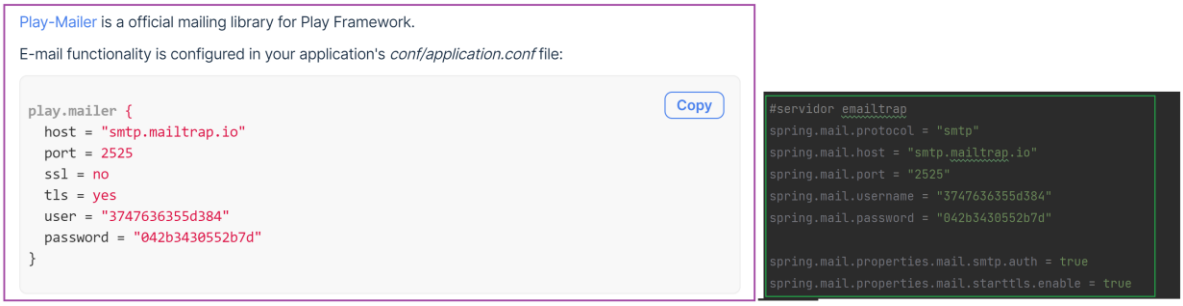
Testando do serviço de email.

Para enviar emails é necessário algum servidor de email com protocolo SMTP. Então a **nível de teste da pra usar o mailTrap** *(dá pra acessar com a conta do github)*, mas ele nunca vai ser utilizado em produção:



O Spring tem na sua documentação as propriedades que podem ser configuradas para o servidor de email. https://docs.spring.io/spring-boot/docs/current/reference/html/application-properties.html#appendix.application-properties.mail :

A ideia **é configurar as propriedades do servidor de email** com as propriedades que o **mailTrap** oferece para a gente:



Outro ponto importante é que o Spring oferece um starter para lidar com emails, então é preciso importa-lo (caso queira utilizar o módulo de email do spring):

```xml
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-mail</artifactId>
</dependency>
```

Agora usando as classes **fornecidas pelo starter de email do Spring** dá pra implementar o método de envio, com um objeto que representa o email e um objeto que de fato o envia:

```java
@Service
public class EmailServiceImpl implements EmailService {

    @Autowired
    private JavaMailSender mailSender;

    @Override
    public void enviarEmails(String mensagem, List<String> emails) {

        String [] emailsDosUsuarios = emails.toArray(new String[emails.size()]);

        SimpleMailMessage mensagemEmail = new SimpleMailMessage();

        mensagemEmail.setFrom("todoGaming@hotmail.com.br");
        mensagemEmail.setSubject("Meta no dia final de conclusão");
        mensagemEmail.setText(mensagem);
        mensagemEmail.setTo(emailsDosUsuarios);

        mailSender.send(mensagemEmail);
    }
}
```

Usando o CommandLineRunner pra enviar os emails de forma manual dá pra verificar se o serviço está funcionando corretamente:

```java
@Autowired
private EmailService emailService;

public static void main(String[] args) {
    SpringApplication.run(TodoApplication.class, args);
}

@Bean
public CommandLineRunner runner(){
    return args ->{

        List<String> emails = Arrays.asList("d933dd8ddf-df0d77@inbox.mailtrap.io");
        emailService.enviarEmails( mensagem: "Testando serviço de emails", emails);
        System.out.println("emails enviados");
    };
}
```