DaaSIoT Spring Boot Module

Version: 0.0.2
Group ID: `it.sebyone`
Artifact ID: `daasiot-spring-module`
License: MIT
Maintainer: SebyOne Srl

Overview

Il modulo **DaaSIoT Spring Boot** fornisce un **Bean Spring preconfigurato** che gestisce in modo automatico il ciclo di vita di un **nodo DaaS** basato sulla libreria nativa `libdaas`, attraverso il wrapper Java `daasiot-java`.

Ogni applicazione Spring Boot che include questo modulo diventa automaticamente un **nodo DaaS** nel network, connesso agli altri nodi tramite le API JNI.

Caratteristiche principali

- Inizializzazione automatica del nodo DaaS tramite `DaasAgentService`.
- Configurazione centralizzata tramite `application.yml`.
- Gestione completa del ciclo di vita (`init → perform → end`).
- Supporto per driver, mappe e modalità operative DaaS.
- Endpoint REST di test e diagnostica integrati.
- Pubblicato come pacchetto Maven Central.

Struttura del modulo

```
it/sebyone/daasiot/spring/
 ├── config/
 │   └── DaasProperties.java         # Configurazione Spring (mappa daas: in YAML)
 ├── service/
 │   └── DaasAgentService.java       # Gestione del nodo DaaS (ciclo di vita e
API)
 ├── controller/
 │   └── DaasController.java         # Endpoint REST di test
 └── autoconfigure/
     └── DaasAutoConfiguration.java  # Auto-configurazione Spring Boot
```

Funzioni principali del servizio

`DaasAgentService`

Bean Spring che incapsula la logica nativa di `DaasAPI` e gestisce un singleton per processo.

Metodo	Descrizione
`init()`	Inizializza il nodo con `doInit()` e `doPerform()`.
`shutdown()`	Arresta il nodo con `doEnd()` e chiude `libdaas`.
`getStatus()`	Restituisce lo stato del nodo (`NodeState`).
`reset()`	Esegue `doReset()` sul nodo.
`mapNode(long remoteDin)`	Mappa un nodo remoto nella rete DaaS.
`locate(long remoteDin)`	Localizza un nodo remoto nella rete DaaS.

Il servizio è automaticamente caricato da Spring se la proprietà `daas.enable=true`.

🌐 Endpoint REST di test

Il modulo include un **controller preconfigurato** (`DaasController`) per testare il funzionamento del nodo.

Metodo	Endpoint	Descrizione	Esempio output
`GET`	`/daas/status`	Mostra lo stato corrente del nodo	`[DaaS] Node Status: NODE_STATE_OK`
`GET`	`/daas/reset`	Esegue un reset del nodo	`[DaaS] Node reset executed.`
`GET`	`/daas/map?remoteDin=102`	Mappa un nodo remoto	`[DaaS] map(102) -> DAAS_ERR_NONE`
`GET`	`/daas/locate?remoteDin=102`	Localizza un nodo remoto	`[DaaS] locate(102) -> DAAS_ERR_NONE`
`POST`	`/daas/push?remoteDin=102`	Pusha il DDO passato nel body	`[DaaS] push(102, ddo) -> DAAS_ERR_NONE`
`POST`	`/daas/pull?remoteDin=102`	Riceve l'eventuale DDO dal nodo remoto	`[DaaS] pull(102, ddo) -> DDO Object`
`GET`	`/daas/shutdown`	Arresta e chiude il nodo	`[DaaS] Node shutdown completed.`

⚙️ Configurazione (`application.yml`)

Esempio completo:

```yaml
daas:
  enable: true
  nodeName: "SpringNode"
  din: 101
  sid: 100
```

```
  list_drivers:
    - link_type: "LINK_INET4"   # o valore numerico 2
      uri: "127.0.0.1:2020"

  list_maps:
    - remote_din: 102
      link_type: "LINK_INET4"
      uri: "127.0.0.1:2021"
```

Proprietà supportate

Chiave	Tipo	Descrizione
`daas.enable`	boolean	Abilita il nodo DaaS
`daas.nodeName`	string	Nome logico del nodo
`daas.din`	int	Node DIN (Device ID)
`daas.sid`	int	SID
`daas.list_drivers[].link_type`	enum/long	Tipo di driver (`LINK_INET4`, `LINK_BT`, ecc.)
`daas.list_drivers[].uri`	string	URI locale del driver
`daas.list_maps[].remote_din`	int	DIN remoto da mappare
`daas.list_maps[].link_type`	enum/long	Tipo di link
`daas.list_maps[].uri`	string	URI remoto del nodo

🧠 Utilizzo come dipendenza Maven

Aggiungi la seguente dipendenza nel tuo `pom.xml`:

```
<dependency>
    <groupId>it.sebyone</groupId>
    <artifactId>daasiot-spring-module</artifactId>
    <version>0.0.4</version>
</dependency>
```

Spring Boot caricherà automaticamente il bean `DaasAgentService` grazie all'auto-configurazione (`DaasAutoConfiguration`).

Non è richiesta alcuna istanziazione manuale:

```
@Service
public class MyService {
    private final DaasAgentService daas;
```

```
    public MyService(DaasAgentService daas) {
        this.daas = daas;
    }

    public void printStatus() {
        System.out.println("Nodo DaaS: " + daas.getStatus());
    }
}
```

🧩 Esecuzione locale

1. Compila il progetto:

   ```
   mvn clean package
   ```

2. Esegui l'app:

   ```
   mvn spring-boot:run
   ```

   oppure:

   ```
   java -jar target/daasiot-spring-module-0.0.2.jar
   ```

3. Testa gli endpoint:

   ```
   http://localhost:8080/daas/status
   http://localhost:8080/daas/reset
   http://localhost:8080/daas/map?remoteDin=102
   ```

👨‍💻 Sviluppo e contatti

Organization: SebyOne Srl
Project: DaasIoT
Maintainer: support@sebyone.it
