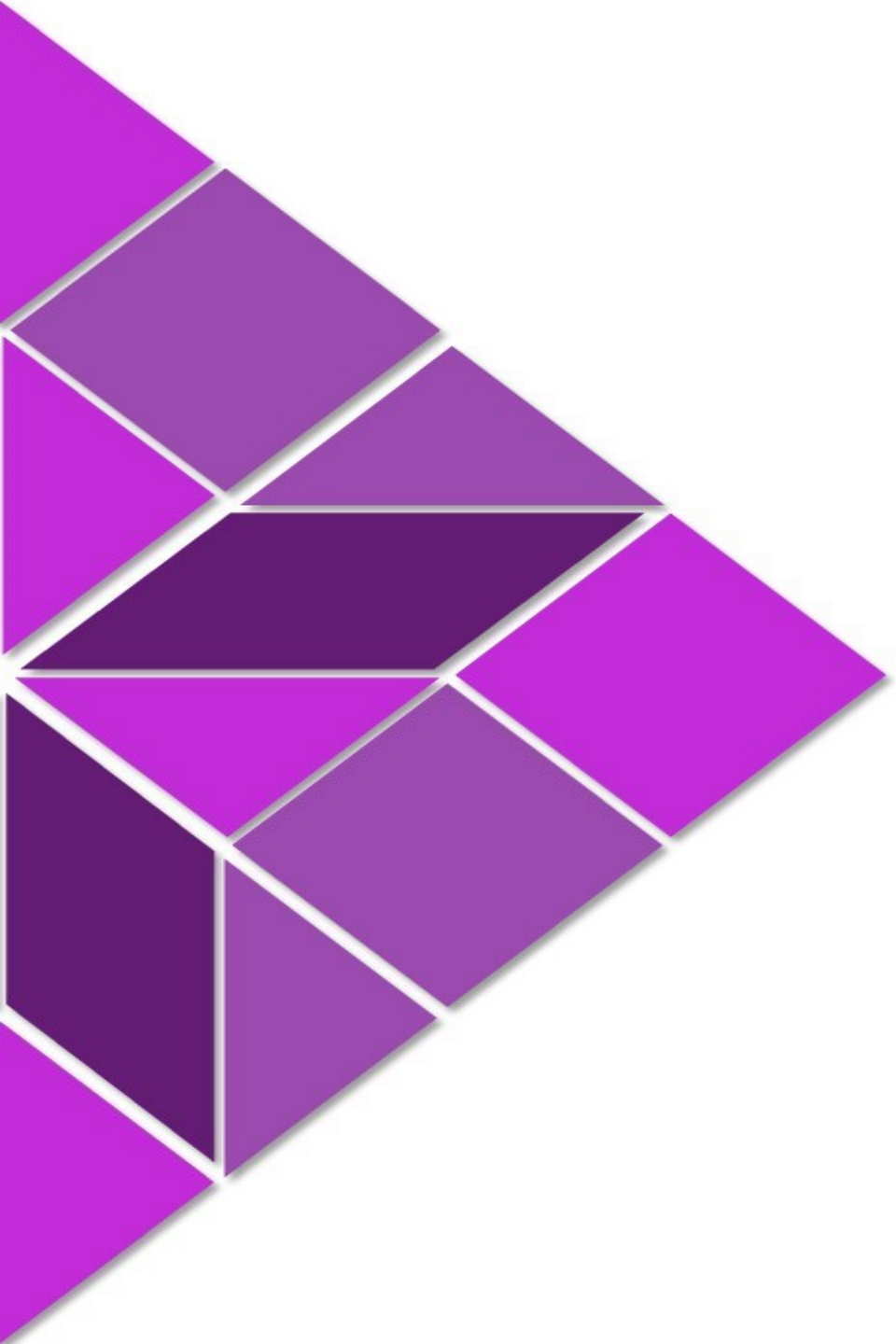


Appendix-JNDI and Datasource

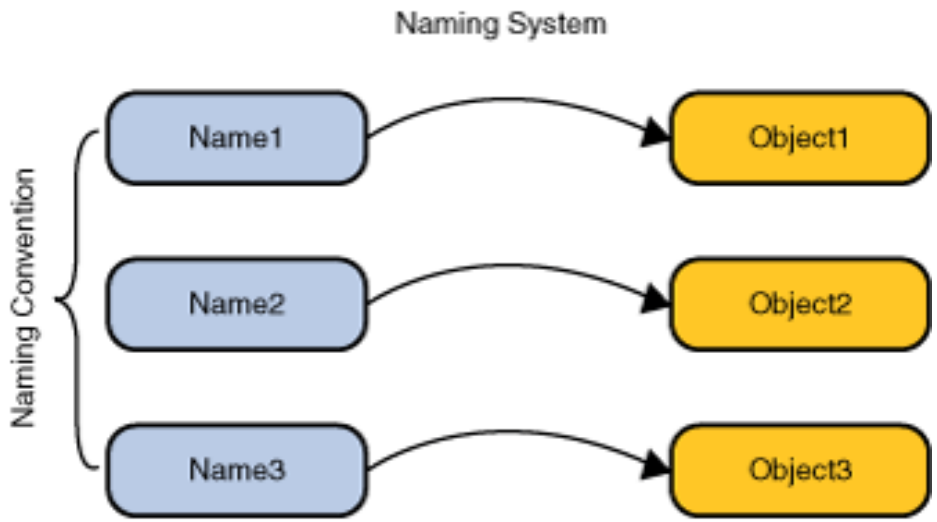
Presented by Hosein Zare
Twitter: @zare88biz

JNDI

- Java Naming And Directory Interface یک API میباشد که به برنامه های نوشته شده توسط جاوا قابلیت نامگذاری و طبقه بندی میدهد
- این روش به برنامه شما قابلیت decoupling میدهد و به شما اجازه میدهد سیستم های ماژولار طراحی کنید

مفاهیم

- مفهوم Naming
  - به معنای نامگذاری هر سرویس (یا شئ) میباشد
  - این مفهوم به طور کلی در سیستم های کامپیوتری مشاهده میشود به طور مثال
    - آدرس یک ایمیل (example@email.com)
    - فایل سیستم ها (/usr/bin/vlc)
    - LDAP (cn=Rosanna Lee, o=Sun, c=US)

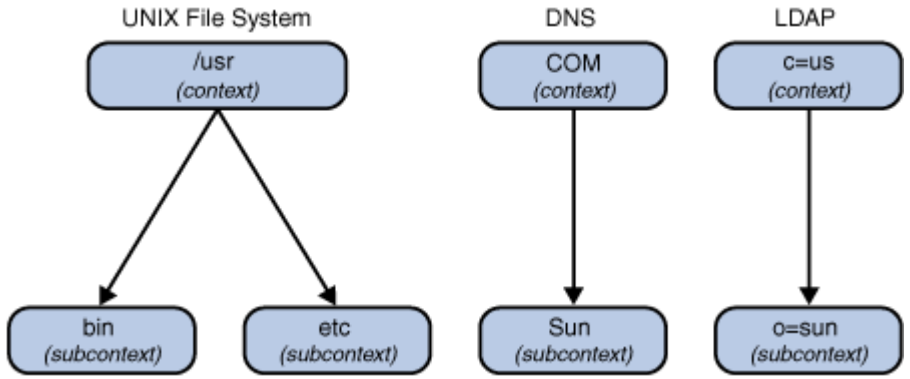


مفاهیم

- مفهوم Bindings
  - به ارتباط دادن بین یک نام و یک شئ binding میگویند
  - مثلاً یک آدرس DNS شامل binding میباشد که آن را به یک ip در شبکه نگاشت میدهد
- مفهوم References and Addresses
  - در سرویس naming هیچگاه کپی از اشیاء نگهداری نمیشوند بلکه reference یا pointer از آن ذخیره میشود
  - آدرس ها اطلاعاتی در رابطه با چگونگی دسترسی به اشیاء استفاده میشود

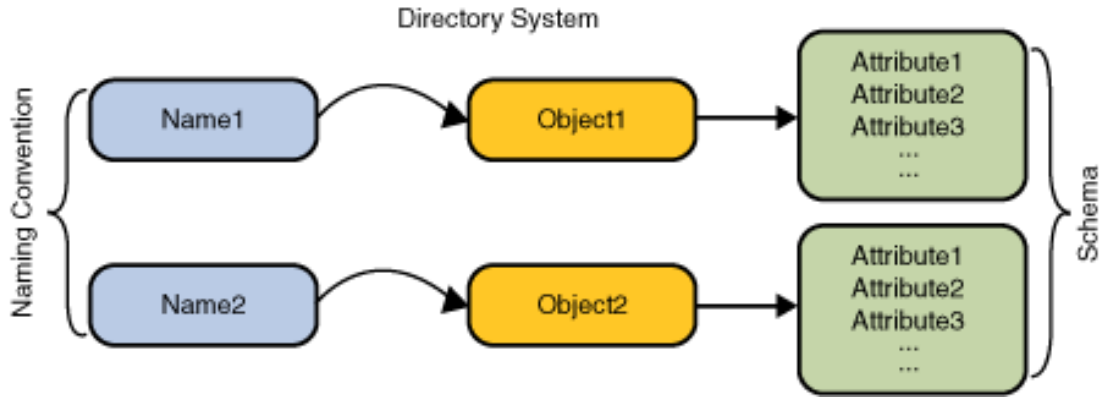
مفاهیم

- مفهوم Context
  - به مجموعه ای از binding های بین نام ها و اشیاء گفته میشود
  - Context همیشه در خود قابلیتی به نام lookup دارد که به شما امکان پیدا کردن شئ توسط نام را مشخص میکند
  - یک نام در یک context میتواند به یک context دیگر در زیرمجموعه خود bind شود (subcontext)



مفاهیم

- مفهوم Directory Service
  - خیلی از naming service ها از directory service ها مشتق شده اند
  - Directory service یک نام را به یک شئ به علاوه ی attribute هایش نگاشت میدهد
  - هر attribute خود شامل کلید و مقدار میباشد
  - در این سیستم شما برای lookup کردن شئ مورد نظر باید از یک search روی attribute ها نیز استفاده کنید

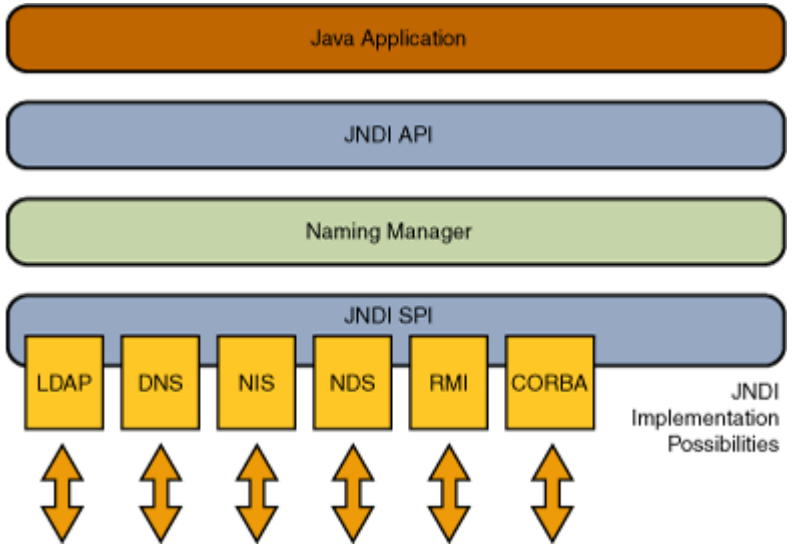


مفاهیم

- مفهوم Directory
  - یک directory مجموعه از directory object ها را به هم متصل میکند
- ترکیب Naming و Directory سرویس ها
  - Directory ها معمولاً اشیاء خود را به شکل درختی نگهداری میکنند.
  - در این حالت directory خود نقش naming context ها را بازی میکنند به علاوه اینکه نگهدارنده ی attribute ها هم هستند

معماری JNDI

- معماری JNDI شامل یک API و یک SPI (service provider interface) میباشد. برنامه های جاوایی از JNDI برای دسترسی به سرویس های مختلف استفاده میکنند.
- SPI این امکان را میدهد که سرویس های مختلفی را به سیستم اضافه کرده تا برنامه های دیگر توسط JNDI به آنها دسترسی پیدا کنند



Packages :

- `javax.naming`
- `javax.naming.directory`
- `javax.naming.ldap`
- `javax.naming.event`
- `javax.naming.spi`

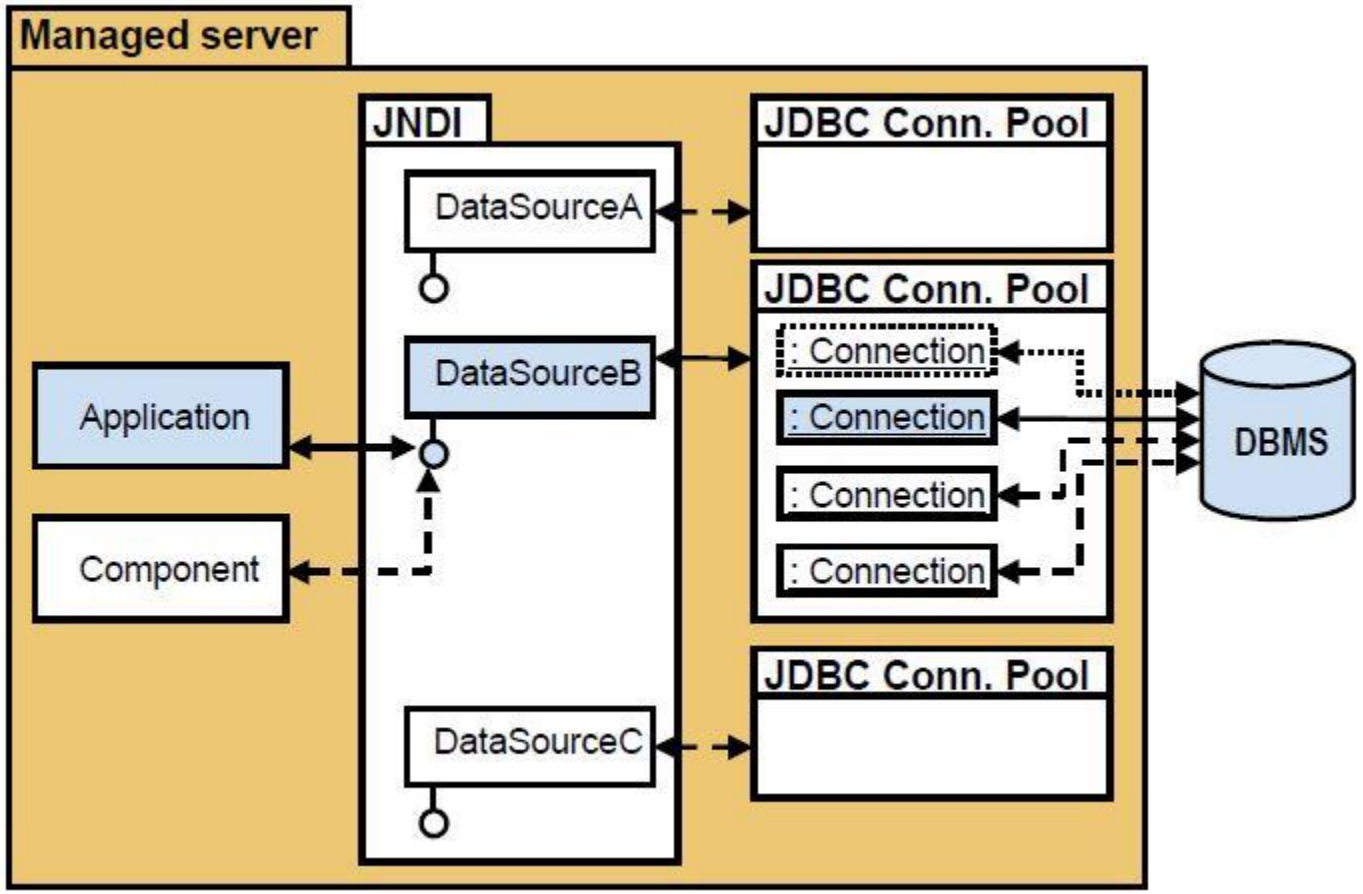
javax.sql.DataSource

- به طور کلی DataSource به شئ اطلاق میشود که به یک منبعی از داده دسترسی دارد
- که میتواند دستیابی به یک سرور پیچیده پایگاه داده باشد یا حتی دسترسی به یک منبع اطلاعاتی درون یک فایل
- DataSource میتواند به شکل یک object factory که به مجموعه از connection ها ارجاع دارد استفاده شود
- از آن به عنوان جایگزینی از DriverManager استفاده میشود
- DataSource ها با JNDI میتوانند کار کنند به طوری که بتوان آنها را از یک context lookup کرد و استفاده کرد.

Database Connection Pool

- Connection pool یک سری connection به پایگاه داده cache شده است که به شما اجازه میدهد در سیستم های بزرگ با تعداد درخواست بالا ، از آنها استفاده مجدد کنید
- استفاده از این روش به بهینه سازی استفاده از حافظه موقت و شبکه کمک میکند.

Databse Connection Pool



Lookup Datasource

- برای lookup کردن data source از طریق JNDI باید از قبل آن را در application server یا web server خود تعریف کرده باشید . سپس از طریق کد زیر میتوانید آن را پیدا کنید و از آن استفاده نمایید

```
// Obtain our environment naming context
Context envCtx = (Context) InitialContext.doLookup("java:comp/env");
// Look up our data source
DataSource ds = (DataSource) envCtx.lookup("jdbc/hrDS");
// Allocate and use a connection from the pool
Connection conn = ds.getConnection();
//... use this connection to access the database ...
conn.close();
```