## LL gramatika

```
startump_seq <stat seq>
<stat class> --> @startnote <multiple token> <stat class>
<stat class> --> @package word <color> <stat class>
<stat class> --> @packageend word <stat class>
<stat class> --> object word <stat class>
<stat class> --> word <relation> <arrow> <relation> word <action> <stat class>
<stat class> --> enduml_class
<stat class> --> @interface word { <attrs> <stat class>
<stat class> --> @enum word { <enum > <stat class>
<stat class> --> @class <abstract> word { <attrs> <stat class>
<color> --> e
<color> --> color t
<relation> --> relation
<relation> --> e
<arrow> --> AGREGATION
<arrow> --> GENERALIZATION
<arrow> --> COMPOSITION
<arrow> --> ASSOCIATION
<action> --> @startaction <multiple_token>
<action> --> e
<enum> --> word <enum>
<enum> --> }
<abstract> --> abstract
<abstract> --> e
<attrs> --> <static> @attribute <mod> word word <attrs>
<attrs> --> <static> @method <mod> word() word <startmth> <attrs>
<attrs> --> }
<startmth> --> @startmth <msg>
<startmth> --> e
```

```
<static> --> STATIC
<static> --> e
<mod> --> private
<mod> --> protected
<mod> --> public
<mod> --> package_mod
<mod> --> e
<stat seq> --> @actor word <stat seq>
<stat seq> --> word <type> word @startactmsg <u u> <stat seq>
<stat seq> --> @startnote <note> <stat seq>
<stat seq> --> continue_line <stat seq>
<stat seq> --> @activate word <stat seq>
<stat seq> --> @deactivate word <stat seq>
<stat seq> --> @start_sep <multiple token> <stat seq>
<stat seq> --> @altstart @condstart <multiple token> <stat seq>
<stat_seq> --> @else @condstart <multiple_token> <\stat_seq>
<stat_seq> --> @altend <\stat_seq>
<stat seq> --> @boxstart @namestart <multiple token> <color> <stat seq>
<stat seq> --> @boxend <stat seq>
<stat seq> --> == <multiple token>
<type> --> request
<type> --> response
<multiple token> --> word <multiple token>
<multiple token> --> PARAM TO END
```