TABELL 1.

Funktion	*Modultyper*	*Förklaring*
`object|mapping remap_url(object request_id);`	URL	Returnerar antingen `request_id` (med iallafall en eller annan variabel modifierad, förhoppningsvis), i vilket fall den används för att göra en ny request (vilket innebär att modulen kommer att anropas igen, se upp!), eller också så kan funktionen returnera en mapping med någon av `http_*` funktionerna (se nedan). Slutligen så kan man returnera noll, i vilket fall allt bara fortsätter.

TABELL 1.

Funktion	*Modultyper*	*Förklaring*
`object|mapping find_file(string file_name, object request_id);`	Location	Returnera ett öppet filobjekt (see uLPC dokumentation om /precompiled/file) om det finns en fil som matchar file_name som den här modulen hanterar, eller en response mapping (lämpligen genererad med `http_*` funktionerana) om man inte vill att extension moduler ska användas igenom. Mappingen kan vara mycket praktisk om man vill returnera att användaren ska skriva in ett lösenord, eller en redirect till en ny fil. En bra funktion att använda i den här funktionen kan vara `open(filename, mode)`, som returnerar just ett sådant öppet filobjekt. Observera att om modulen har location (`query_location` returnerar) ”`/foo/`”, och filen ”`/foo/bar/`” efterfrågas, kommer location modulen att få ”`bar`/” som första argument. `request_id` finns dokumenterad i en annan artikel i denna tidning, den om skript i spinner, liksom response mapping.
`array find_dir(string dir_name, object request_id);`	Location	Returnera en array med filnamn eller noll. `dir_name` som `file_name` hos `find_file`. Det finns en default `find_dir` i `module.lpc` som returnerar 0. Så om man inte vill returnera några directorylistningar kan man bara låta bli att defeniera den här funktionen.
`array stat_file(string file_name, object request_id);`	Location	`file_name` som hos `find_file`. Returnera resultatet av en `file_stat` (se uLPCdokumentationen) eller noll. Om man vill kan man skippa även den här funktionen, directorymodulen och en del andra moduler kan dock kräva att den finns för att din modul ska fungera precis som ett ”vanligt” filsystem.
`string real_file(string file_name, object request_id);`	Location	Returnera vad filen `file_name` egentligen heter (t.ex om modulen sune hittar sina filer i `/usr/www/foo/`, och filen `bar` efterfrågas i den, så ska den returnera `/usr/www/foo/bar`). Den här metoden är inte alls nödvändig, det kan ju t.ex. hända att dina filer egentligen inte existerar på disken, men den snabbar upp saker som cgimoduler och uLPC skript.
`mapping query_tag_callers();` `mapping query_container_callers();`	Parser	Anropas från main_parsern för att registrera en parse modul. Returvärdet är en mapping enligt: `([ ”tag”:funktion, ... ])`, eller en tom mapping `([])`.

TABELL 1.

Funktion	*Modultyper*	*Förklaring*

`array (string) query_extensions();`
`array (string) query_file_extensions();`
	Extension	Returnera en array med extensions som modulen tänker hantera. Lämpligtvis implementeras enligt följande:
		`array (string) query_extensions()`
		`{`
		`  return query("extensions");`
		`}`
		(förutsatt att variabeln extensions är definierad, som en `TYPE_STRING_LIST`, se avsnittet om variabler senare).

`mapping handle_extension(string extension, object request_id);` `mapping handle_file_extension(object file, string ext, object request_id);`	Extension	Returnera antingen ett resultat eller 0 (noll). Anropas av Spinner när modulen ska hantera en extension, i fallet `handle_extension` innan en location modul har hittat en fil som passar (används knappast alls, den enda modulen som finns som använder det är explicit timestamp modulen, som returnerar last modification date (mtime) på en fil, om man lägger till .timestamp till filnamnet.) Vanligtvis gör man file_extension moduler, då är det det senare fallet som gäller. 'file' är en klon av `/precompiled/file`, se uLPC dokumentationen. Kortfattat kan man nämna att 'file' har metoden `read()`, kan vara lämpligt i det här fallet.
`mapping first_try(object request_id);`	First	Används bland annat av logger.lpc, den där modulen som loggar requests i användarnas egna directoryn på Lysator. Om man returnerar en response mapping så kommer det att tas som svaret. Den här funktionen anropas aldrig vid interna requests, dvs, vid `<insert>` och liknande.
`mapping last_resort(object request_id);`	Last	Används av relay modulen, om man sätter priority till last. Anropas bara om ingen annan modul har lyckats hitta ett matchande dokument, och aldrig vid interna requests.
`string query_location();`	Location	Returnera vilken position i det virtuella filsystemet modulen ska få. Även här använder man lämpligen värdet av en variabel, eftersom användaren då själv kan välja var modulen ska hamna.

TABELL 1.

Funktion	*Modult yper*	*Förklaring*
`void create();`	Alla	Anropas av uLPC automatisk när objektet (instansen av din modul) skapas. Här definierar man lämpligtvis sina modullokala Spinner variabler med hjälp av '`defvar`'.
`void start();`	Alla	Anropas av spinner precis innan modulen ska vara redo att ta emot requests (requests hanteras med de modultypsspecifika callbackfunktionerna).
`string status();`	Alla	Anropas av konfigurationsinterfacet i tid och otid för att få status om modulen. Ska returnera en sträng, beståendes av HTML som tål att stoppas i en `<dd>` i en `<dl>` lista.
`string info();`	Alla	Om man defenierar den här funktionen används resultatet istället för element tre i det som `register_module()` returnerar för att beskriva modulen i konfigurationsinterfacet.
`string|void check_variable(string variable, mixed will_be_set_to);`	Alla	Om man behöver kontrollera värdet hos variabler innan de sätts defenitivt så gör man lämpligtvis det i den här funktionen. Oftast behöver man ju dock inte kontrollera variablers giltighet, eftersom alla värden som man kan sätta dem till är giltiga. Mer information om spinnervariabler i allmänhet kommer senare.
`string query_name();`	Alla	Ska returnera namnet på modulen, används istället för element två i arrayen som `register_module()` returnerar för att beskriva modulen i konfigurationsinterfacet. Kan vara mycket användbart om man kan ha mer än en kopia av en modul. Användaren kan dock själv sätta vilket namn som helst på modulen i konfigurationsinterfacet.

`array auth(array from);`
`array userinfo(string username);`
`array userlist();`
`array user_from_uid(int uid);`
	Auth	Se userdb.lpc modulen. Oftast behöver man inte implementera en egen användardatabas och authentifieringsmodul, och det hör inte till grunderna i spinnerprogrammering.

`mapping parse_directory(object request_id);`	Directories	Givet en intern url (i `request_id->not_query`) ska den här funktionen returnera en directorylistning i HTML.