OAT REST

	Endpunkt	Methode	Request Body	Body Notiz	Sample Output	Category 2
▼ telescope						

	/telescope/info	GET			{
					"result": [
					"OpenAstroTracker",
					"V1.9.05",
					"+49*16",
					"169*07",
					"01/01/21",
					"04:22:58"
					],
					"status": "success"
					}
						telescope

	/telescope/status	GET			{
					"result": {
					"declination": "+900000",
					"rightAscension": "221334",
					"isSlewing": false,
					"isTracking": true,
					"status": "Tracking"
					},
					"status": "success"
					}
						telescope

	/telescope/position	GET			{
					"result": {
					"declination": "+90*00'00",
					"rightAscension": "22:13:34"
					},
					"status": "success"
					}
						telescope

	/telescope/datetime	GET			{
					"result": {
					"currentDate": "27/05/22",
					"currentTime": "10:53:18",
					"currentUTCOffset": "-02"
					},
					"status": "success"
					}
						telescope

	/telescope/datetime	POST	{
			"utcTimestamp": "1653647354",
			"utcOffset": "-120"
			}
				Für Javascript UTC Timestamp durch 1000 teilen
				Offset ist in JS schon richtig
				(Ich kann dir auch nen Parameter “jsTimestamp” geben und mach das Backend seitig)
					{
					"result": "successfully executed :SC27/05/22#:SL10:29:14#:SG-2.0# to set telescope date and time.",
					"status": "success"
					}
						telescope

	/telescope/geolocation	GET			{
					"result": {
					"lat": "+49*16",
					"lng": "30*28"
					},
					"status": "success"
					}
						telescope

	/telescope/geolocation	POST	{
			"lat": "+49*16",
			"lng": "169*07"
			}
				Keine Ahnung wie wir da die Eingabe machen wollen. Hier können wir die Umrechnung auf das komische OAT Format mit * auch im Backend machen	{
					"result": "successfully executed St+49*16#:Sg169*07# to set telescope geolocation.",
					"status": "success"
					}
						telescope

	/telescope/action	POST	{
			“action”: “setHome”
			}
			{
			“action”: “reset”
			}
			{
			“action”: “toggleParking”
			}
			{
			“action”: “togglePrecision”
			}
			{
			“action”: “toggleTracking”
			}
					{
					"result": "successfully executed action toggleParking on telescope.",
					"status": "success"
					}
						telescope

	/telescope/slew	POST	{
			“to”: “home”
			}
			{
			“to”: “target”
			}
					{
					"result": "slewing telescope to home.",
					"status": "success"
					}
						telescope

	/telescope/move	POST	{
			"direction": "w"
			}
			{
			"direction": "n"
			}
			{
			"direction": "s"
			}
			{
			"direction": “e”
			}
				Direction = Himmelsrichtung auf englisch	{
					"result": "moving telescope into direction e",
					"status": "success"
					}
						telescope

	/telescope/move/quit	POST	{
			"direction": "w"
			}
			{
			"direction": "n"
			}
			{
			"direction": "s"
			}
			{
			"direction": “e”
			}
			{
			"direction": “a”
			}
				Direction a = alles stoppen	{
					"result": "stopping movement of telescope into direction e.",
					"status": "success"
					}
						telescope

	/telescope/slew/rate	POST	{
			"speed": 1
			}
				Speed geht von 1 (langsam) bis 4 (schnell)	{
					"result": "setting slew speed to 1.",
					"status": "success"
					}
						telescope

▼ devices						

	/devices	GET			{
					"result": [
					"/dev/cu.EMBERTON",
					"/dev/cu.Bluetooth-Incoming-Port",
					"/dev/cu.usbserial-0001"
					],
					"status": "success"
					}
						devices

	/devices	POST	{
			"comDevice": "/dev/cu.usbserial-10",
			"baudRate": 19200
			}
				Hier können wir nochmal drüber nachdenken ob wir das nicht über einen Settings Endpunkt Speichern wollen.
				Der USB Serial Port wird sich beim Raspi wenn er am Tracker hängt ja nichtmehr ändern.
						devices

▼ target						

	/target/position	GET			{
					"result": {
					"declination": "+90*00'00",
					"rightAscension": "22:13:34"
					},
					"status": "success"
					}
						target

	/target/position	POST	{
			"declination": "+90*00'00",
			"rightAscension": “22:10:00”
			}
					{
					"result": "setting target position to DEC: +90*00'00 RA: 22:10:00.",
					"status": "success"
					}
						target