Building Structure Arrays

You can build structures in two ways:

- Using assignment statements
- Using the `struct` function

Building Structure Arrays Using Assignment Statements

You can build a simple 1−by−1 structure array by assigning data to individual fields. MATLAB automatically builds the structure as you go along. For example, create the 1−by−1 `patient` structure array shown at the beginning of this section:

```
patient.name = 'John Doe';
patient.billing = 127.00;
patient.test = [79 75 73; 180 178 177.5; 220 210 205];
```

Now entering

```
patient
```

at the command line results in

```
name: 'John Doe'
billing: 127
test: [3x3 double]
```

`patient` is an array containing a structure with three fields. To expand the structure array, add subscripts after the structure name:

```
patient(2).name = 'Ann Lane';
patient(2).billing = 28.50;
patient(2).test = [68 70 68; 118 118 119; 172 170 169];
```

The `patient` structure array now has size `[1 2]`. Note that once a structure array contains more than a single element, MATLAB does not display individual field contents when you type the array name. Instead, it shows a summary of the kind of information the structure contains:

```
patient
patient =
1x2 struct array with fields:
    name
    billing
    test
```

You can also use the `fieldnames` function to obtain this information. `fieldnames` returns a cell array of strings containing field names.

As you expand the structure, MATLAB fills in unspecified fields with empty matrices so that

- All structures in the array have the same number of fields.
- All fields have the same field names.

For example, entering `patient(3).name = 'Alan Johnson'` expands the `patient` array to size `[1 3]`. Now both `patient(3).billing` and `patient(3).test` contain empty matrices.

> **Note** Field sizes do not have to conform for every element in an array. In the `patient` example, the `name` fields can have different lengths, the `test` fields can be arrays of different sizes, and so on.

Building Structure Arrays Using the struct Function

You can preallocate an array of structures with the `struct` function. Its basic form is

```
strArray = struct('field1',val1,'field2',val2, ...)
```

where the arguments are field names and their corresponding values. A field value can be a single value, represented by any MATLAB data construct, or a cell array of values. All field values in the argument list must be of the same scale (single value or cell array).

You can use different methods for preallocating structure arrays. These methods differ in the way in which the structure fields are initialized. As an example, consider the allocation of a 1−by−3 structure array, `weather`, with the structure fields `temp` and `rainfall`. Three different methods for allocating such an array are shown in this table.

Method	Syntax	Initialization
`struct`	`weather(3) = struct('temp', 72, ... 'rainfall', 0.0);`	`weather(3)` is initialized with the field values shown. The fields for the other structures in the array, `weather(1)` and `weather(2)`, are initialized to the empty matrix.
`struct` with `repmat`	`weather = repmat(struct('temp', ... 72, 'rainfall', 0.0), 1, 3);`	All structures in the `weather` array are initialized using one set of field values.
`struct` with cell array syntax	`weather = ... struct('temp', {68, 80, 72}, ... 'rainfall', {0.2, 0.4, 0.0});`	The structures in the `weather` array are initialized with distinct field values specified with cell arrays.

Memory Requirements for Structures

You do not necessarily need a contiguous block of memory to store a structure. The memory for each field in the structure needs to be contiguous, but not the entire structure itself.